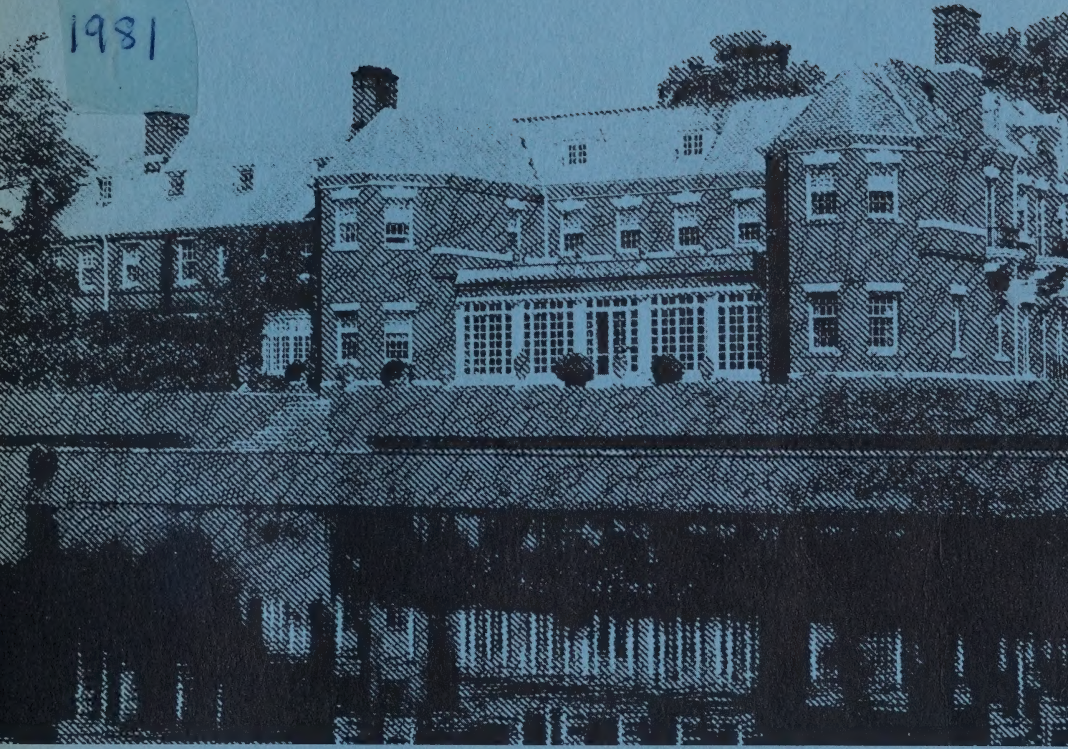
621.3811
Al5
1981



Proceedings

NINETEENTH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

September 30, October 1-2, 1981

Allerton House, Monticello, Illinois
Sponsored by the
Department of Electrical Engineering and the
Coordinated Science Laboratory of the
University of Illinois at Urbana-Champaign

The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.

To renew call Telephone Center, 333-8400

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

ENGINEERING

APR 09 REC'D

JUL 05 1988

JUN 07 REC'D

L161—O-1096

PROCEEDINGS

NINETEENTH ANNUAL ALLERTON CONFERENCE ON COMMUNICATION, CONTROL, AND COMPUTING

W. K. Jenkins
D. V. Sarwate
Conference Co-Chairmen

Conference held
September 30-October 2, 1981
Allerton House
Monticello, Illinois

Sponsored by
The Department of Electrical Engineering
and
The Coordinated Science Laboratory
of the
UNIVERSITY OF ILLINOIS
at
Urbana-Champaign

621.3811
Al5
1981

Engin

FOREWORD

The Nineteenth Annual Allerton Conference on Communication, Control anc Computing was held at Allerton House, Monticello, Illinois on September 30, October 1, and October 2, 1981. The conference was sponsored by the Department of Electrical Engineering and the Coordinated Science Laboratory of the University of Illinois at Urbana-Champaign. These PROCEEDINGS contain the papers presented at the conference.

In addition to the large numbers of contributed papers, there were two organized sessions of invited papers. We are grateful to Professor Max Mintz of the University of Pennsylvania for organizing and chairing a session on Fire Control Systems, and to Professor Ümit Özgüner of Ohio State University for organizing and chairing a session on Large Scale Systems. Their time and efforts on behalf of the Conference are deeply appreciated. We would also like to express our gratitude to the other session chairmen and authors for their contributions to the success of the Conference. The musical contributions of Professor Thomas Siwe and his Steel Drum Band to the social program are also gratefully acknowledged.

The support and encouragement of Professor G. W. Swenson, Jr., Head of the Department of Electrical Engineering, and Professor R. T. Chien, Director of the Coordinated Science Laboratory are appreciated and acknowledged. We would also like to thank our colleagues, Professors J. A. Abraham, M. T. Basar, T. Ü. Basar, D. J. Brown, J. B. Cruz, Jr., B. E. Hajek, I. N. Hajj, P. V. Kokotovic, D. P. Looze, M. C. Loui, D. E. Muller, D. C. Munson, Jr., J. H. Patel, W. R. Perkins, H. V. Poor, F. P. Preparata, M. B. Pursley, P. W. Sauer, M. S. Schlansker, S. Swamy, T. N. Trick, and M. E. Van Valkenburg for their generous assistance in organizing the Conference.

We are grateful to Mr. Harold Lawler and Mr. Steven Williams for their handling of the business affairs of the Conference. Finally, very special thanks are due to Ms. Rose Harris, Mrs. Rosemary Wegeng and Mrs. Phyllis Young for their gracious and impeccable assistance in all aspects of the organization of the Conference, from the initial preparation of the Conference announcement through Conference registration to the final editing of these PROCEEDINGS.

W. K. Jenkins

D. V. Sarwate

TABLE OF CONTENTS

FOREWORD iii

I-A: ALGORITHMICS I: Computational Geometry

Chairman: D. J. Brown
University of Illinois

MANHATTAN WIRING 1
R. Raghavan, J. Cohoon and S. Sahni

SQUEEZING CONSTANT FACTORS OF GEOMETRIC ALGORITHMS 11
J. L. Bentley

A LINEAR-TIME ALGORITHM FOR DECOMPOSING RECTILINEAR STAR-SHAPED POLYGONS INTO CONVEX QUADRILATERALS ... 21
J.-R. Sack and G. T. Toussaint

THE P-CENTER PROBLEM IN THE PLANE IS NP-COMPLETE .. 31
A. Marchetti-Spaccamela

COMPUTATIONAL GEOMETRY ON INTEGER GRIDS 41
J. M. Keil and D. G. Kirkpatrick

I-B: SOURCE CODING AND QUANTIZATION

Chairman: D. L. Snyder
Washington University, St. Louis

DPCM ENCODING OF REGENERATIVE COMPOSITE SOURCES ... 51
M. J. Carter and D. L. Neuhoff

ADAPTIVE COMPANDED PULSE-CODE-MODULATION 60
J. A. Bucklew and F. T. Tzeng

AN EXPERIMENTAL APPROACH FOR DESIGNING VARIABLE-LENGTH TO VARIABLE-LENGTH CODES 69
N. R. Kolb and J. G. Dunham

INFORMATION THEORY AND CAUSAL INFORMATION TRANSMISSION WITH FEEDBACK 70
T. Berger and G. Munson

SUFFICIENCY, AUTOREGRESSION, AND TIME SERIES MODELING 78
B. W. Dickinson

CONVERGENCE OF VECTOR QUANTIZERS WITH APPLICATION TO THE DESIGN OF OPTIMAL QUANTIZERS 79
E. F. Abaya and G. L. Wise

k-DIMENSIONAL POLAR QUANTIZERS FOR GAUSSIAN SOURCES 89
P. F. Swaszek and J. B. Thomas

SOME RECENT RESULTS ON VECTOR QUANTIZING.......... 98
N. J. A. Sloane

I-C: FILTERING

Chairman: N. C. Gallagher, Jr.
Purdue University

SOME RESULTS ON THE MEDIAN FILTERING OF SIGNALS AND ADDITIVE WHITE NOISE.................. 99
T. A. Nodes and N. C. Gallagher, Jr.

AN ADAPTIVE FILTER USING POLYNOMIAL LEAST SQUARES METHODS............................ 109
G. H. Xia and A. R. Stubberud

OPTIMUM TIME VARYING FILTERS FOR TRANSIENT WAVEFORM ESTIMATION............................... 119
K. B. Yu and C. D. McGillem

VOLTERRA SERIES APPROACH TO FILTERING............. 128
J. T.-H. Lo and S.-K. Ng

CONVERGENCE OF STOCHASTIC APPROXIMATION ALGORITHMS WITH LINEAR FILTERS................... 138
H. Huang

I-D: LARGE SCALE SYSTEMS AND MODEL SIMPLIFICATION

Chairman: D. P. Looze
University of Illinois

ON THE DECOUPLING OF LINEAR SYSTEMS INTO SINGLE INPUT-MULTIPLE OUTPUT SUBSYSTEMS.......... 144
H. Y. Kim and E. Y. Shapiro

ON DECOUPLING OF MULTIVARIABLE SQUARE PLANTS... 153
M. Eslami and M. D. Lodaya

ON THE SELECTION OF THE EIGENVALUES TO BE RETAINED IN A REDUCED-ORDER MODEL................ 163
G. Zhao, P. Rozsa and N. K. Sinha

INSENSITIVE REDUCED-ORDER MODELS................. 174
M. T. Tran and M. E. Sawan

GENERIC POLE ASSIGNMENT USING DYNAMIC OUTPUT FEEDBACK.................................. 175
T. E. Djaferis

CONTINUOUS-TIME STOCHASTIC MODEL SIMPLIFICATION..................................... 185
J. K. Tugnait

II-A: ALGORITHMICS II: Parallel Computation and VLSI

Chairman: J. L. Bentley
Carnegie-Mellon University

LABELLING AND ADDRESSING IN DISTRIBUTED SYSTEMS.......... 195
N. Santoro and R. Khatib

DESIGNING EFFICIENT PARALLEL ALGORITHMS FOR MULTIPLE-OUTPUT COMPUTATIONS WITH FAN-IN CONSTRAINTS.......... 204
N. Santoro and R. L. Probert

A PARALLEL COMPUTATION OF CONNECTED COMPONENTS.... 211
G. D. Lakhani

A PARALLEL ALGORITHM FOR DETERMINING CONVEX HULLS OF SETS OF POINTS IN TWO DIMENSIONS.......... 214
A. L. Chow

I/O LIMITATIONS ON MULTI-CHIP VLSI SYSTEMS........ 224
M. Snir

AREA-TIME BOUNDS FOR VLSI CIRCUITS.......... 234
K. Wada, K. Hagihara and N. Tokura

A VLSI ALGORITHM AND ARRAY FOR THE QR-METHOD...... 235
L. Johnsson

II-B: COMMUNICATION SYSTEMS

Chairman: T. Ü. Basar
University of Illinois

AVERAGE ERROR PROBABILITY FOR DS-SSMA COMMUNICATIONS: THE GRAM-CHARLIER EXPANSION APPROACH.......... 237
K.-T. Wu

A SIMPLIFIED APPROACH TO THE DESIGN AND GENERATION OF SPREAD-SPECTRUM SIGNALS.......... 247
G. R. Cooper and E. J. Heald

ERROR PROBABILITY BOUNDS FOR UNBALANCED QAM OVER THE GAUSSIAN PHASE JITTERED, DISPERSIVE CHANNEL.......... 248
T. L. Lim

ADAPTIVE EQUALIZATION AND PHASE TRACKING FOR SIMULTANEOUS ANALOG/DIGITAL DATA TRANSMISSION..... 249
T. L. Lim and M. S. Mueller

CORRELATION TECHNIQUES APPLIED TO THE BIT SYNCHRONIZATION PROBLEM.......... 259
C. R. Guarino

A TYPE II HYBRID ARQ SCHEME FOR DATA TRANSMISSION IN MOBILE RADIO SYSTEMS.............. 260
R. A. Comroe and D. J. Costello, Jr.

OPTIMAL CODING FOR INFORMATION TRANSMISSION THROUGH A POISSON TYPE CHANNEL.................................... 261
A. A. Lazar and S. C. Schwartz

II-C: STOCHASTIC CONTROL

Chairman: M. T. Basar
University of Illinois

STOCHASTIC CONTROL OF TWO COMPETING QUEUES........ 269
J. S. Baras and S. A. Belbas

OPTIMAL CONTROL OF A M/M/1 QUEUE.................. 279
A. A. Lazar

A CRAMER-RAO TYPE LOWER BOUND ON ESTIMATION PERFORMANCE FOR STOCHASTIC CONTROL-SYSTEMS........ 290
C.-W. Chen and D. L. Snyder

A MODEL FOR UNCOORDINATED DISTRIBUTED COMPUTATION OF FIXED POINTS....................... 291
D. P. Bertsekas

II-D: CONTROL SYSTEMS

Chairman: J. B. Cruz, Jr.
University of Illinois

FEEDBACK STABILIZATION AND CONTROL OF LINEAR NEUTRAL SYSTEMS............................ 296
T. J. Tarn, M. W. Spong and K. Ito

CONTROLLABILITY OF LINEAR SYSTEMS IN THE PRESENCE OF NONSEPARABLE UNCERTAINTY.............. 304
B. R. Barmish

COMPUTATIONAL ASPECTS OF DISCRETE TIME ADAPTIVE CONTROL................................ 312
R. Cristi and R. V. Monopoli

CONSTRAINED EIGENVALUE/EIGENVECTOR ASSIGNMENT APPLICATION TO FLIGHT CONTROL SYSTEMS................................. 320
E. Y. Shapiro and J. C. Chung

MODELING PROCEDURES FOR HANDLING QUALITIES EVALUATION OF FLEXIBLE AIRCRAFT.................. 329
K. S. Govindaraj, B. J. Eulrich and C. R. Chalk

APPLICATION OF MAXIMUM LIKELIHOOD ESTIMATION TO THE IDENTIFICATION OF THE STABILITY DERIVATIVES OF A WIDE BODY TRANSPORT AIRCRAFT.................................. 339
K. M. Sobel and E. Y. Shapiro

III-A: ALGORITHMICS III: Data Structures

Chairman: D. E. Muller
University of Illinois

EXQUISIT: APPLYING QUICKSORT TO EXTERNAL FILES.... 348
H. W. Six and L. Wegner

TWO VERY FAST TRIE DATA STRUCTURES............... 355
D. E. Willard

ANALYSIS OF A RANDOMIZED DATA STRUCTURE FOR REPRESENTING ORDERED SETS.................... 364
J. L. Bentley, D. F. Stanat and J. M. Steele

DELETING THE ROOT OF A HEAP...................... 373
E.-E. Doberkat

ON THE AVERAGE-CASE COMPLEXITY OF 'BUCKETING' ALGORITHMS........................... 381
S. G. Akl and H. Meijer

QUERY PERFORMANCE OF A MANY-DIMENSIONAL BEST-MATCH ALGORITHM............................ 389
M. R. Dunlavey

III-B: COMMUNICATION NETWORKS

Chairman: B. E. Hajek
University of Illinois

AN APPLICATION OF GROUP TESTING TO THE DESIGN OF MULTI-USER ACCESS PROTOCOLS.................. 397
D. F. Towsley and J. K. Wolf

CONFLICT RESOLUTION PROTOCOLS FOR RANDOM MULTIPLE-ACCESS CHANNELS WITH BINARY FEEDBACK.. 404
T. Berger and N. Mehravari

DISTRIBUTED ASSIGNMENT OF LINKS IN MOBILE RADIO NETWORKS.................................. 412
D. J. Baker, A. Ephremides, J. Weiselthier

ANALYSIS OF THE PACKET FORMATION PROCESS IN PACKET-SWITCHED NETWORKS...................... 413
J. S. Meditch

DISTRIBUTED MANAGEMENT SYSTEM OF CRIPTOGRAPHIC KEYS IN COMMUNICATION NETWORKS 421
E. Okamoto, K. Nakamura, Y. Shimizu, Y. Teshigawara, and Y. Sato

III-C: OPTIMAL CONTROL

Chairman: W. R. Perkins
University of Illinois

THE RICCATI EQUATION, IMPRIMITIVE ACTIONS AND SYMPLECTIC FORMS 430
M. R. Garzia, K. A. Loparo, and C. F. Martin

OPTIMAL QUANTIZED CONTROL 440
T. R. Fischer

A STABLE CONTROLLER ARISING FROM MINIMIZATION OF A QUADRATIC PERFORMANCE CRITERION WITH AN INDEFINITE STATE WEIGHTING MATRIX 450
A. K. Tugcu and R. E. Reid

A HIERARCHICAL OPTIMAL FINAL-VALUE CONTROL STRATEGY FOR NONLINEAR SYSTEMS REALIZABLE WITH SMALL COMPUTERS 451
B. Asselmeyer

III-D: SYSTEMS

Chairman: P. V. Kokotovic
University of Illinois

A SERVO COMPENSATOR DESIGN APPROACH FOR VARIABLE STRUCTURE SYSTEMS 452
A. J. Calise and K. V. Raman

EIGENSTRUCTURE ASSIGNMENT WITH MINIMUM POLE SENSITIVITY TO PLANT PARAMETER VARIATIONS 461
S. H. Lee

TRANSFORMATION OF NONHOMOGENEOUS NONLINEAR SYSTEMS 462
R. Su, G. Meyer, and L. R. Hunt

SOME RESULTS ON STABILITY AND STABILIZATION OF SYSTEMS WITH RETARDATION 463
A. Thowsen

A Z-TRANSFORM THEORY FOR DISTRIBUTED SENSING AND CONTROL 470
S. S. L. Chang

AN EFFICIENT METHOD FOR ANALYZING A CLASS OF NONLINEAR SEMISTATE EQUATIONS 480
M. Saidahmed and M. E. Zaghloul

III-E: MULTIPROCESSING

Chairman: M. S. Schlansker
University of Illinois

OPTIMAL SCHEDULING OF ASSEMBLY LANGUAGE KERNELS FOR VECTOR PROCESSORS.................... 486
S. Arya and D. A. Calahan

DESIGN OF A COMMUNICATIONS KERNEL FOR LOOSELY COUPLED MULTIPROCESSORS.................. 494
A. Asthana, C. S. Roberts and G. K. Swanson

APPROXIMATE ANALYTICAL PERFORMANCE EVALUATION IN BUS ORIENTED MULTICOMPUTER SYSTEMS............ 504
S. V. Iyengar and L. L. Kinney

BOUNDS ON THE PARALLEL PROCESSOR SPEEDUP.......... 514
U. Banerjee

CENTRALLY CONTROLLED SEGMENTED BUS ARCHITECTURE....................................... 524
E. T. Fathi and M. Krieger

A FLEXIBLE PACKET-SWITCHING CHIP SET AND ITS APPLICATIONS................................. 534
M. E. Brown and B. W. Weide

APPLICATION OF 2-D BIN PACKING ALGORITHMS FOR TASK SCHEDULING IN PASM.......................... 542
D. L. Tuomenoksa and H. J. Siegel

IV-A: ALGORITHMICS IV: Interconnection Networks

Chairman: C. Kruskal
University of Illinois

EXPECTED CAPACITY OF BINOMIAL CONCENTRATORS....... 543
G. M. Masson and S. B. Morris

SOME PROPERTIES OF THE BINARY N-CUBE AS A NETWORK INTERCONNECTION STRUCTURE................ 553
J. G. Kuhl and S. M. Reddy

ON GRAPH THEORETIC APPROACH TO n- AND (2n-1) STAGE INTERCONNECTION NETWORKS.................. 559
D. P. Agrawal

IV-B: DETECTION

Chairman: H. V. Poor
University of Illinois

AN OBSERVATION CONCERNING SIGNAL TO NOISE RATIO PROPERTIES OF CONTINUOUS AND DISCRETE TIME DETECTORS.................................. 569
D. R. Halverson and G. L. Wise

FINITE LENGTH DISCRETE MATCHED FILTERS........... 576
A. B. Martinez and J. B. Thomas

ROBUST MULTIPLE-INPUT MATCHED FILTERS............ 586
C.-T. Chen and S. A. Kassam

A COMPARISON OF POLARITY COINCIDENCE CORRELATION WITH UNCLIPPED DETECTORS FOR CORRELATED NOISE INPUTS... 596
M. Kanefsky and A. Q. K. Rajput

DECISION MAKING WITH LIKELIHOOD BASED LOWER PROBABILITIES.............................. 605
T. L. Fine and M. Wolfenson

DETECTION OF CONSTANT SIGNALS IN SKEWED NOISE..... 607
Y. F. Huang and J. B. Thomas

CASE STUDY OF LIKELIHOOD RATIO DETECTOR.......... 617
C.-C. Lee

IV-C: FOUNDATIONS OF COMPUTING

Chairman: M. C. Loui
University of Illinois

STEADY-PACED-OUTPUT AND FRACTIONAL-ON-LINE ALGORITHMS ON A RAM............................. 618
G. K. Manacher

A HARD PROBLEM FOR NTIME(n^d)..................... 626
L. Denenberg and H. R. Lewis

TWO IMPLEMENTATIONS OF RECURSIVE PATH ORDERING ON MONADIC TERMS........................ 634
P. Lescanne

POLYNOMIAL AVERAGE-TIME SATISFIABILITY PROBLEMS.. 644
P. W. Purdom, Jr. and C. A. Brown

ON UNIVERSALITY OF CONCURRENT EXPRESSIONS WITH SYNCHRONIZATION PRIMITIVES................. 645
T. Ito and Y. Nishitani

IV-D: FIRE CONTROL SYSTEMS

Organizer
and Chairman: M. Mintz
University of Pennsylvania

CONTROL PROBLEMS IN THE FIELD ARTILLERY SYSTEM.... 654
S. S. Wolff

MICROPROCESSOR-BASED OPTIMAL CONTROLLERS FOR A HELICOPTER TURRET CONTROL SYSTEM................. 659
N. Coleman and N. K. Loh

CONCEPTS FOR IMPROVED GUN FIRE CONTROL SYSTEMS 669
H. H. Burke

A DESIGN METHODOLOGY FOR ESTIMATORS AND PREDICTORS IN FIRE CONTROL SYSTEMS 678
J. F. Leathrum

ON DESIGNING ROBUST PREDICTORS BASED ON FINITE STATE MARKOV CHAINS: A GAME THEORETIC APPROACH.... 688
E. B. Pate, M. Mintz, W. Dziwak, and S. A. Goodman

IV-E: FAULT-TOLERANCE AND RELIABILITY

Chairman: J. H. Patel
University of Illinois

A NEW APPROACH TO SYSTEM DIAGNOSIS 697
K. Nakajima

ON TESTING STUCK-OPEN FAULTS IN CMOS COMBINATIONAL CIRCUITS 707
R. Chandramouli

A DEDUCTIVE METHOD FOR THE SIMULATION OF FAULTS IN PROGRAMMABLE LOGIC ARRAYS 713
F. Özgüner

MORPHIC PROPERTIES OF DETERMINISTIC AND NONDETERMINISTIC FAULT MODELS 721
B. L. Havlicsek and G. G. L. Meyer

MODELING AND RELIABILITY OF MULTI-MINIPROCESSOR SYSTEMS 722
K. Konstantinides

V-A: ALGORITHMICS V: Combinatorial Computing

Chairman: E. M. Reingold
University of Illinois

ON A SCHEDULING PROBLEM WITH TWO TYPES OF MACHINES 731
K. Nakajima and S. L. Hakimi

SOME NP-COMPLETE PROBLEMS ON GRAPH DECOMPOSITIONS 741
C. J. Colbourn

SHIFTING ALGORITHMS FOR TREE PARTITIONING WITH GENERAL WEIGHTING FUNCTIONS 746
R. I. Becker and Y. Perl

CUTWIDTH PROBLEMS IN GRAPHS 752
E. M. Gurari and I. H. Sudborough

POLYNOMIAL CONSTRUCTION OF OPTIMAL PREFIX TABLES FOR TEXT COMPRESSION...................... 762
Y. Choueka, A. S. Fraenkel and Y. Perl

V-B: POWER SYSTEMS

Chairman: P. W. Sauer
University of Illinois

DECENTRALIZED LOAD FLOW ALGORITHM FOR LARGE-SCALE POWER SYSTEM......................... 769
H. Li, M. H. Mickle and W. G. Vogt

REDUCTION OF HIGH ORDER SYSTEM WITH APPLICATION TO CONTROLLER DESIGN................. 777
I. El-Nahas, N. K. Sinha and R. T. H. Alden

AN EXACT Y-Δ TRANSFORMATION FOR THE $\hat{S}$-$\hat{E}$ PRESENTATION OF ELECTRIC POWER SYSTEMS........... 786
M. Ilic-Spong, J.-B. Liu and J. Zaborszky

AN EFFICIENT METHOD OF SOLUTION FOR NEARLY SYMMETRIC MATRICES................................ 796
Y. F. Zhou

V-C: LARGE SCALE SYSTEMS

Organizer
and Chairman: Ü. Özgüner
Ohio State University

DECENTRALIZED DYNAMIC CLEARING OF CONGESTED MULTI-DESTINATION NETWORKS....................... 797
P. E. Sarachik

ON THE DECENTRALIZED CONTROL OF AN AUTOMATED HIGHWAY.. 804
R. E. Magee and R. E. Fenton

COMMUNICATION CONTROL FOR LOCAL NETWORKS.......... 814
R. S. Printis

DISTRIBUTED REAL-TIME SIMULATION OF MULTIBODY SYSTEMS... 819
H. Hemami, Y.-F. Zheng and P. A. Grygier

EXTENSIONS OF MULTIVARIABLE CONTROL SYSTEM DESIGN TECHNIQUES TO LARGE SCALE SYSTEMS.......... 829
W. S. Levine and A. Leung

DECENTRALIZED REGULATION OF CELLULAR STRUCTURES... 836
Ü. Özgüner and B. Yildrim

V-D: DIGITAL IMAGE AND SIGNAL PROCESSING

Chairman: D. C. Munson, Jr.
University of Illinois

NOISE REJECTION IN ARRAY DATA.................... 842
M. Simaan

FILTERING OF NOISY IMAGES USING MARKOV RANDOM FIELD MODELS.............................. 850
R. L. Kashyap and R. Chellappa

AN EFFICIENT SOURCE CODING TECHNIQUE FOR DATA COMPRESSION OF MULTILEVEL DIGITIZED IMAGES........ 860
C. B. Fong and M. Kanefsky

IMAGE SOURCE CODING USING MEDIAN FILTER ROOTS.................................... 869
G. R. Arce and N. C. Gallagher, Jr.

C-MATRIX TRANSFORM................................ 879
R. Srinivasan and K. R. Rao

SYSTOLIC AND SIMD ALGORITHMS FOR DIGITAL FILTERING................................ 880
M. A. Yoder and L. J. Siegel

ESTIMATING GENERAL TWO-DIMENSIONAL MOTION FROM IMAGE SEQUENCES............................ 890
Y. P. Hsu and T. S. Huang

BLOCK TRUNCATION CODING ON PASM.................. 891
L. J. Siegel, E. J. Delp, T. N. Mudge, and H. J. Siegel

MANHATTAN WIRING*

RAGHUNATH RAGHAVAN,
Sperry-Univac
Roseville, MN 55164

JAMES COHOON and SARTAJ SAHNI
University of Minnesota
Minneapolis, MN 55455

ABSTRACT

The Manhattan wiring of point pairs is considered. We develop an $O(n^3)$ algorithm to determine if a set of n point pairs can be wired on a single layer; when this is possible, our algorithm generates the layout. Determining the minimum number of layers needed to wire a set of n point pairs is shown to be NP-hard.

1. INTRODUCTION

In the most general wire routing case, the conductor paths can be curves. However, in automated systems, conductor paths are usually constrained by the fabrication equipment to contain only horizontal and vertical segments.

Wiring where each path is made up of horizontal and vertical segments only is called rectilinear wiring. It is often thought of as wiring on a grid. All wire end points are grid points; each conductor path is constrained to lie along grid points only; and conductor paths are not allowed to cross. The separation between grid lines reflects the mininum conductor separation necessary to avoid inductive cross-talk. Figure 1 shows an example of wiring on a grid.

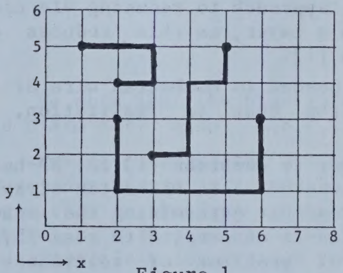


Figure 1

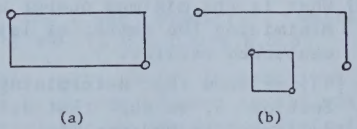

(a)
(b)

Figure 2

Any point on a grid may be specified by its x- and y-coordinates. Assuming that consecutive grid lines are one unit apart, the coordinates of grid points are natural numbers. Let $S = \{(u_i, v_i, x_i, y_i) \mid 1 \leq i \leq n\}$ be a set of n point pairs to be connected. The point (u_i, v_i) is to be connected to the point (x_i, y_i), $1 \leq i \leq n$. S is called a wire set, since each point pair defines a wire.

There are a number of applications where it is necessary to limit the number of bends on each conductor. For example, when fabricating microwave integrated circuits, the conductors, called microstrip lines, function as transmission lines. Bends on a microstrip line result in reflections, and this reduces the transmission efficiency of the line.

So, an important problem we consider is that of wiring S on a grid subject to the added constraint that no wire may bend more than k times. This problem has been studied, for example, by [3]. When the wiring is

---

* This research was supported in part by the National Science Foundation under Grant MCS 80-005856.

restricted such that no wire has more than one bend, it is called Manhattan wiring.

With the constraint that conductor paths are not allowed to cross, it should be clear that not all wire sets S can be wired on a single layer. Hence, we need to consider multilayered boards. A multilayered board has k layers for wiring. Since the points to be connected represent module locations, pins from the modules run vertically through all the layers. There may also be plated-through holes called vias at other locations. Each layer on which wiring is permitted is a grid, and the same restrictions apply as before.

Usually, the cost of realization goes up sharply with the number of layers used for wiring. So it is desirable to minimize the number of layers used. Also, engineering considerations might limit the number of layers available to do the wiring. For example, integrated circuits can often have just 3 or 4 layers of metallization.

In Section 2, we show that it is possible to efficiently determine whether a wire set S is Manhattan wireable on one layer. Our algorithm has time complexity $O(n^3)$, where $n = |S|$. For wire sets S that are Manhattan wireable on one layer, our algorithm also obtains the layout in the stated time bound. For those wire sets that cannot be Manhattan wired on one layer, the following questions are of interest:

(1) What is the maximum number of wires of S that can be Manhattan wired on one layer?
This question is related to minimizing via usage. Using vias typically lowers system reliability. An approach to reducing via usage is to maximize the number of wires on a layer, as this reduces the need for feedthroughs to other layers [1].

(2) What is the minimum number of layers needed to Manhattan wire S?
Minimizing the number of layers cuts the cost of realization, as mentioned earlier.

In [4], we show that determining the answer to question (1) is NP-hard. In Section 3, we show that determining whether S is Manhattan wireable on two layers is NP-complete. This implies that determining the answer to question (2) is NP-hard. The reader is encouraged to read [5], a recent survey of the complexity of several problems of relevance to design automation.

2. MANHATTAN WIRING ON ONE LAYER

In this section, we present an algorithm to determine whether a given wire set S can be Manhattan wired feasibly on one layer. In general, there are two ways to lay out a wire in Manhattan fashion, as shown in Figure 2(a). However, the presence of other wires may preclude one or even both these layouts (see Figure 2(b)).

Let $T \subseteq S$ be the set of wires for which there is only one feasible layout. The wires in T are forced, i.e., their layouts are the same in all feasible layouts for S. Let $A=(a,b,c,d)$ and $B=(e,f,g,h)$ be two wires in S. Any one of the following relations may hold between A and B:

(1) The two wires cannot be laid out feasibly on the same layer. See Figure 3(a). Clearly, the wiring is infeasible.

(2) Either layout for each wire does not directly influence the layout for the other. See Figure 3(b). We denote this as $(a,b,c,d)I(e,f,g,h)$ (i.e., they are pairwise independent).

(3) The upper layout for (a,b,c,d) forces the (upper) layout for (e,f,g,h); however, the lower layout for (a,b,c,d) does not influence the layout for (e,f,g,h) in any way. See Figure 3(c). We denote this situation as (a,b,c,d)U(e,f,g,h).

(4) There is the analogous situaion where only the lower layout for (a,b,c,d) forces the (lower) layout for (e,f,g,h). This situation is denoted as (a,b,c,d)L(e,f,g,h).

(5) Finally, there is the situation where the upper (lower) layout for (a,b,c,d) forces the lower (upper) layout for (e,f,g,h). See Figure 3(d). In this case, we say that the two wires are pairwise dependent, and write it as (a,b,c,d)D(e,f,g,h).

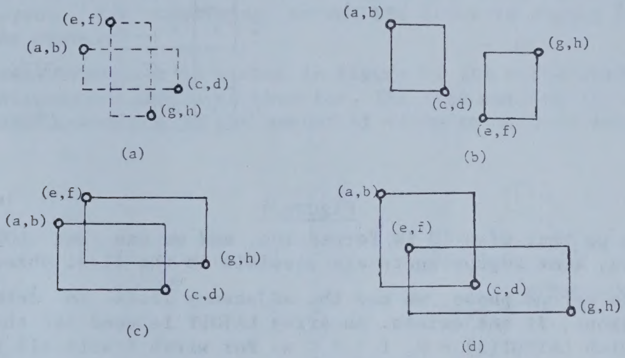


Figure 3

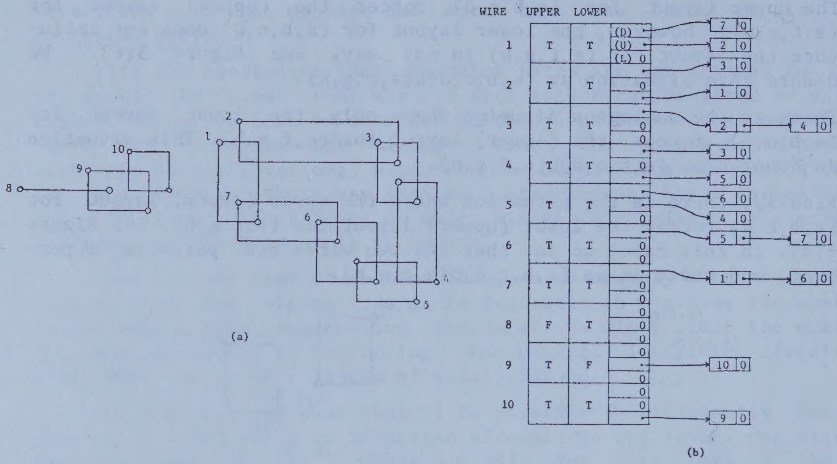
We note that the binary relations D and I are symmetric, while L and U are not. Also, (a,b,c,d)L(e,f,g,h) $\Leftrightarrow$ (e,f,g,h)U(a,b,c,d)

Having encapsulated all the possible wire pair interactions in these four binary relations L, U, D, and I, we now proceed to the wiring algorithm. In the first phase of the algorithm, we consider all $n(n-1)/2$ (where $n = |S|$) pairs of wires in S. For each pair (A,B), we determine the appropriate interaction relation (i.e., ADB, AIB, AUB, ALB, infeasible). If it is determined that the wires A and B cannot be wired feasibly , the algorithm terminates. Also, during this phase, certain wires are identified as forced.

The output from the first phase is a set of adjacency lists, three for each wire in S. The three lists associated with any wire A in S are called its D-list, its U-list and its L-list. Let $X \in \{D, U, L\}$. A's X-list contains all wires B in S such that the relation AXB holds.

Associated with each wire A in S, there are two Boolean variables, UPPER and LOWER. UPPER(A) (LOWER(A)) is true if and only if the upper (lower) wiring path for A is currently usable, and is false otherwise. For the wires that have been identified as forced, one of UPPER(), LOWER() will be false. Note that in any feasible layout, we can never have wires with both UPPER() and LOWER() false. Figure 4 shows a data structure that incorporates the adjacency lists and Boolean variables. Figure 4(a) shows a wire set S, and Figure 4(b) shows this data structure after the first phase.

Note that all the forced wires have not been detected. For example, suppose the wire pair (9,10) is examined after the wire pair (8,9). Since we know that wire 9 is forced by wire 8 to use the upper path,



(a) (b)

Figure 4

9U10 tells us that wire 10 is forced too, and we can set LOWER(10) to false. Thus, some improvements are possible in the first phase.

In the second phase, we use the adjacency lists to determine the actual layout, if one exists. An array LAYOUT is used for this purpose. We begin with LAYOUT(i) = 0, $1 \le i \le n$. For wires i with all three adjacency lists empty, we set LAYOUT(i) to ´U´ if UPPER(i) is true, and ´L´ if not. Thus, in Figure 4, LAYOUT(8) = ´L´. Clearly, such wires, when laid out, do not interfere with each other, i.e., they cannot intersect.

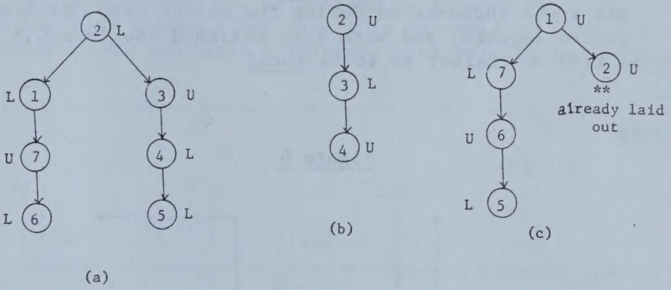
To lay out the remaining wires, we need to conduct a search. We start with any wire j with LAYOUT(j) = 0, preferably one with only one of UPPER(j), LOWER(j) true. We arbitrarily pick one of the layouts in the set { x | x ∈ {UPPER,LOWER} and x(j) = true }. This layout for j forces the layouts for all wires on j´s D- and Z-lists, where Z = U if x = UPPER and Z = L otherwise. These forced layouts, in turn, force the layouts of other wires; and so on. We follow through all such forced layouts, making sure that no layout intersects any other.

If this search over all wires that get forced is successful (i.e., no wire is forced into two different layouts), the layouts for all the wires examined have been determined, and LAYOUT(k) is appropriately set for all wires k in this set. It is clear that the remaining unassigned wires can be assigned independently of the set just assigned; the assignments just made cannot possibly conflict with any choice to be made later. So, if any wire p with LAYOUT(p) = 0 remains, we begin the search again from this wire.

If the search starting with wire j is unsuccessful, the choice of x as the layout for j is infeasible, as it leads to a conflict. x(j) is set to false. If this leads to UPPER(j) = LOWER(j) = false, the wire set S is not Manhattan wireable. If one of UPPER(j), LOWER(j) remains true, this layout is tried for j, and a search is conducted as before. If this too fails, no feasible Manhattan wiring exists for S. In this way, the entire wire set S can be Manhattan wired, if possible.

To illustrate the process, let us complete the example of Figure 4. Starting at wire 9, let us choose the upper layout for it. 9 has no D-list; going along its U-list, we see that 10 is forced into its upper layout. As no other wire is now forced, we set LAYOUT(9) = LAYOUT(10) = ´U´. In the next phase, let us arbitrarily select the lower layout for wire 2. From 2´s adjacency lists, we see that 3 is forced to its upper layout, and 1 to its lower layout. 1 forces 7 up and 3 forces 4 down. 7 then forces 6 down and 4 forces 5 down. The layout of 6 forces 5 up, which conflicts with its previous layout. Figure 5(a) illustrates this situation. So we back up and set LOWER(2) to false, as the lower layout causes infeasiblity down the line. Picking the upper layout for 2 causes the forces shown in Figure 5(b). Since no other wires are forced by this set, we can set LAYOUT(2) = LAYOUT(4) = ´U´ and LAYOUT(3) = ´L´. Finally, we start another search at 1. Choosing the upper layout for 1 causes 7 to be forced down and 2 to be forced up, which agrees with its previous layout. The remaining forces are shown in Figure 5(c). This completes the example.

The formal algorithm is stated in Figure 6. Its correctness follows from the discussion presented thus far. The time complexity is readily seen to be $O(n^3)$, where n is the number of wires to be laid out.



(a) (b) (c)

Figure 5

3. MINIMIZING THE NUMBER OF LAYERS

The problem of determining the minimum number of layers needed to Manhattan wire a wire set S is NP-hard. This result follows from a much stronger result that we obtain in this section; namely, that determining whether S can be Manhattan wired on two layers is NP-complete. We first define the following two problems:

Problem: 2MAN
Input: A wire set S.
Output: ´Yes´ if and only if S can be Manhattan wired on two layers; ´No´ otherwise.

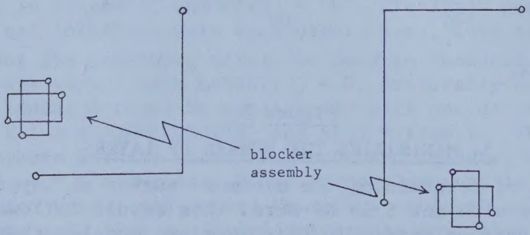
Problem: 3SAT
Input: A set $C = \{C_1, C_2, .. , C_m\}$ of m clauses. Each clause is the disjunction of exactly 3 literals.
Output: ´Yes´ if and only if there is a truth assignment to the variables in the clauses such that every clause is true; ´No´ otherwise.

The 3SAT problem is known to be NP-complete (see [2] for example). 2MAN is readily seen to be solvable in nondeterministic polynomial time. So, to show 2MAN NP-complete, it is sufficient to show that it is NP-hard. We do this by showing 3SAT $\propto$ 2MAN.

Let (A,B) be a wire with two possible layouts. If we are restricted to two layers, then (A,B) can be forced to use one of its two possible layouts by adding a blocker assembly (see Figures 7(a) and (b)).

```
procedure LAY_WIRE(S,n,LAYOUT)
set S; character LAYOUT(n); Boolean UPPER(n), LOWER(n)
for every pair (i,j), $i \neq j$, of wires in S do // Phase 1 //
   determine X such that iXj, $X \in \{D,U,L,I,infeasible\}$, holds
   if X = infeasible then return(´S not feasible´) endif
   insert i and j onto the appropriate D-, L- and U-lists
   update UPPER(i),LOWER(i),UPPER(j) and LOWER(j) as needed
endfor
LAYOUT(i) ← 0, $1 \leq i \leq n$
for i ← 1 to n do // Phase 2 //
   if LAYOUT(i) = 0 then
   L1: if ¬(UPPER(i) or LOWER(i)) then return(´infeasible´) endif
       Let $X \in \{UPPER, LOWER\}$ such that X(i) = true
       Carry out a breadth- or depth-first search starting at wire i
       and visiting each wire whose layout gets forced. This involves
       going through the D-, L- and U-lists of each wire encountered.
       During this search, make sure that infeasible layouts are not
       overlooked.
       if the search is successful then
          all wires encountered during the search are assigned their
          forced layouts, and wire i is assigned the layout X
       else X(i) ← false; go to L1 endif
   endif
endfor
end LAY_WIRE
```

Figure 6



(a) Lower layout forced

(b) Upper layout forced

Figure 7 Blocking a Layout in Two Layers

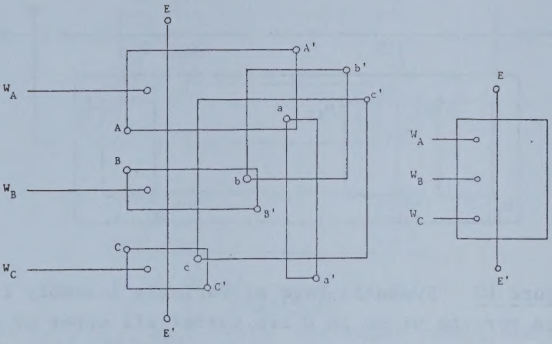
When a layout has been blocked by a blocker assembly, only the remaining feasible layout will be shown.

Several wire constructs are needed for the proof. Figure 8(a) shows a clause assembly. The other endpoint of each of the three wires W_A, W_B and W_C will be specified later. The symbolic form for this assembly is shown in Figure 8(b).

Lemma 1: The wires (a,a´), (b,b´) and (c,c´) of a clause assembly (Figure 8(a)) cannot be laid out on the same layer.

Lemma 2: Any two of the wires (a,a´), (b,b´) and (c,c´) can be laid out on the same layer.

Lemma 3: The set of wires in clause assembly can be laid out on two layers if and only if at least one of $\{W_A,W_B,W_C\}$ is not constrained to lie on the same layer as {(A,A´),(B,B´),(C,C´)}. Note that because of the wire (E,E´), the three wires (A,A´), (B,B´) and (C,C´) must be on the same layer in every two-layer layout of a clause assembly.


E
A'
b'
c'
W_A
a
A
B
W_B
b
B'
C
W_C
c
C'
a'
E'
E
W_A
W_B
W_C
E'

(a) (b)

Figure 8 Clause Assembly

A variable assembly is shown in Figure 9. The broken line is not a part of the assembly and may be ignored at present. m is the number of clauses in the 3SAT instance for wich this assembly is to be used. The variable assembly for variable i is symbolically drawn as in Figure 10.

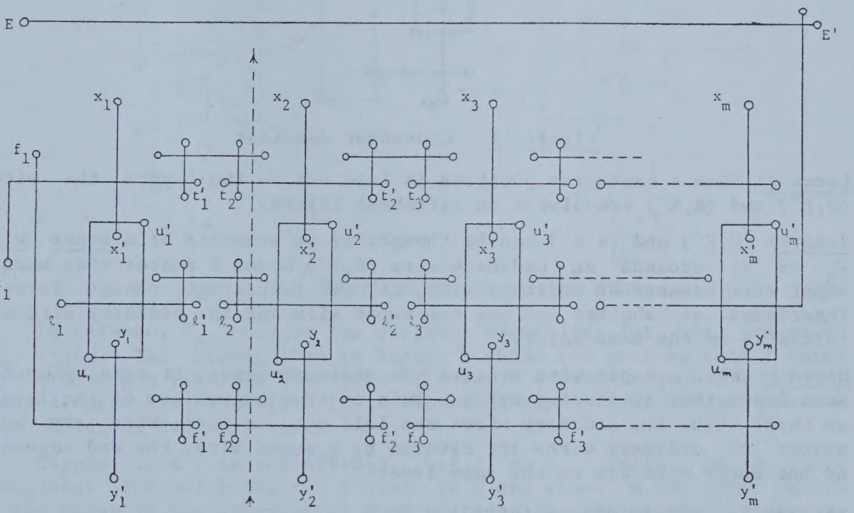

E
E'
x_1
x_2
x_3
x_m
f_1
t'_1
t_2
t'_2
t_3
u'_1
u'_2
u'_3
u'_m
x'_1
x'_2
x'_3
x'_m
t_1
ℓ_1
ℓ'_1
ℓ_2
ℓ'_2
ℓ_3
y_1
y_2
y_3
y_m
u_1
u_2
u_3
u_m
f'_1
f_2
f'_2
f_3
f'_3
y'_1
y'_2
y'_3
y'_m

Figure 9 Variable Assembly

Lemma 4: A variable assembly can be laid out on two layers. Furthermore, every two-layer layout of a variable assembly has the following characteristics:

(i) (E,E') and the wire set $U = \{(u_i,u'_i) \mid 1 \le i \le m\}$ are all forced to be on the same layer.

(ii) All wires in the wire set $T = \{(t_i,t'_i) \mid 1 \le i \le m\}$ are wired on the same layer, and on a different layer from the wire set $F = \{(f_i,f'_i) \mid 1 \le i \le m\}$.

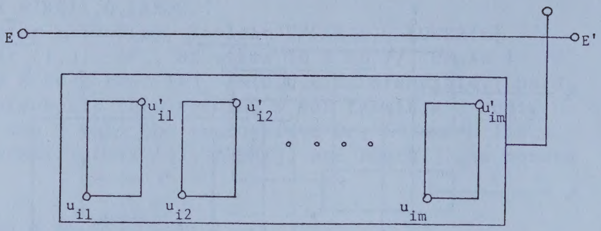


Figure 10 Symbolic Form of Variable Assembly i

(iii)The layouts for the wires in U are either all upper or all lower.

The wire configuration in Figure 11 is a crossover junction.

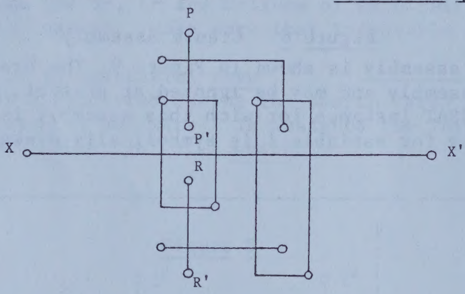


Figure 11 Crossover Junction

Lemma 5: When a crossover junction is laid out on two layers, the wires (P,P´) and (R,R´) are always on different layers.

Remark: (P,P´) and (R,R´) can be thought of as segments of a super wire W_S as it crosses an ordinary wire (X,X´). Lemma 5 states that when a super wire crosses an ordinary wire, it is forced to change layers, regardless of whether or not the super wire and the ordinary wire are initially on the same layer.

Lemma 6: When a super wire crosses two ordinary wires, it ends up on the same layer that it started out on. This is true regardless of the layers on which these two ordinary wires are laid out. In fact, when any even number of ordinary wires are crossed by a super wire, the end segments of the super wire are on the same layer.

Theorem 1: 2MAN is NP-complete.

Proof: As remarked at the start of this section, 2MAN can be solved in nondeterministic polynomial time. So, it is sufficient to show that it is NP-hard. We do this by showing 3SAT $\propto$ 2MAN.

Let $C = \{C_1, C_2, .. , C_m\}$ be any instance of 3SAT. Let the variables in the clauses be v_i, $1 \leq i \leq n$. Corresponding to C, we construct a 2MAN instance as follows:

Step 1 Construct n variable assemblies, all using the same (E,E´) wire (see Figure 12).

Step 2 Place m clause assemblies over the (E,E´) wire by identifying this (E,E´) with the (E,E´) of Figure 8(a).

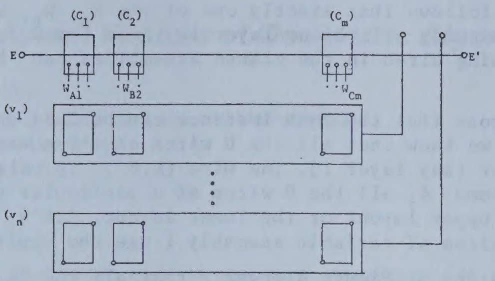


Figure 12

The wires W_{Ai}, W_{Bi}, W_{Ci}, $1 \le i \le m$, are super wires, and correspond to the three literals l_a, l_b, l_c, in clause C_i, i.e., super wire W_{Xi} corresponds to the literal l_x, $\forall X \in \{A,B,C\}$. Let v_x be the variable in literal l_x, i.e., $l_x \in \{v_x, \bar{v}_x\}$, $\forall x \in \{a,b,c\}$. Super wire W_{Xi} originates in the clause assembly for C_i, and terminates in the variable assembly for v_x. If $l_x = v_x$, W_{Xi} terminates as in Figure 13(a). Otherwise, it terminates as in Figure 13(b).

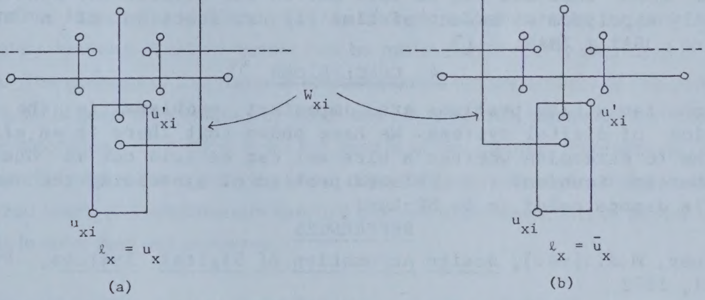


Figure 13 Termination of a Super Wire

In between, W_{Xi} crosses the variable assemblies for the variables v_j, $1 \le j < x$. The broken line in Figure 9 shows the path of such a super wire crossing a variable assembly. It uses a crossover junction as it crosses each ordinary wire. It is seen that super wires always cross an even number of ordinary wires.

Suppose that C is satisfiable. Let $v_i = z_i$, $1 \le i \le n$, be a truth assignment for which all the clauses in C are true. A two-layer layout of the wires in the constructed 2MAN instance is obtained as follows:

(1) (E,E′) and (u_{ij}, u'_{ij}), $1 \le j \le m$, $1 \le i \le n$, are laid out on layer 1. If z_i = true then (d_{ij}, u'_{ij}), $1 \le i \le m$, are laid out using the upper layout. If z_i = false, the lower layout is used. From Lemma 4, it follows that the remaining wires in the variable assemblies get forced onto one or the other of the layers.

(2) Since $v_i = z_i$, $1 \le i \le n$, satisfies all the clauses, at least one literal in every clause is true under this assignment. Hence, the terminating segment (Figure 13(a)) of at least one of the super wires of every clause can be laid out on layer 1. The remaining terminating segments can be laid out on layer 2. From Lemma 6 and the fact that every super wire crosses an even number of ordinary

wires, it follows that exactly one of the W_A, W_B, W_C wires of each clause assembly will be on layer 1. From Lemma 3, it follows that the remaining wires in the clause assemblies can be laid out on layer 2.

Next, suppose that the 2MAN instance can be laid out on two layers. From Lemma 4, we know that all the U wires of all n variable assemblies are on one layer (say layer 1). The wire (E,E') is also on layer 1. Also, from Lemma 4, all the U wires of a particular variable assembly use either the upper layout or the lower layout. Set v_i to true if and only if the U wires of variable assembly i use the upper layout.

An examination of Figure 8 reveals that all the (A,A'), (B,B') and (C,C') wires of the clause assemblies must be on layer 2. From Lemma 3, it follows that the starting segment of at least one of the W_A, W_B, W_C super wires of each clause assembly must be laid out on layer 1. From Lemma 6 and the fact that every super wire crosses an even number of ordinary wires, it follows that the end segment of at least one of the super wires in every clause assembly is laid out on layer 1. From Figure 13, we see that this implies that at least one literal in every clause is true. Hence, the 3SAT instance is satisfiable.

Thus, the 3SAT instance C has answer ´Yes´ if and only if the constructed 2MAN instance can be laid out on two layers. The construction takes only a polynomial amount of time (as a function of n and m). Therefore, 3SAT $\propto$ 2MAN. □

4. CONCLUSIONS

Manhattan wiring problems are important problems in the design automation of digital systems. We have shown that there is an efficient algorithm to determine whether a wire set can be laid out on one layer in Manhattan fashion. A related problem of minimizing the number of layers is demonstrated to be NP-hard.

REFERENCES

[1] Breuer, M.A. [ed.], Design Automation of Digital Systems, Prentice Hall, 1972.

[2] Garey, M.R. and D.S. Johnson, Computers and Intractability - A Guide to the Theory of NP-Completeness, W.H. Freeman and Co., San Francisco, 1979.

[3] Pomentale, T., "An Algorithm for Minimizing Backboard Wiring Functions", Comm. ACM, 8:11, November 1965.

[4] Raghavan, R., J. Cohoon and S. Sahni,"Manhattan and Rectilinear Wiring", Dept. of Computer Science, Univ. of Minnesota, Technical Report 81-5, 1981.

[5] Sahni, S., A. Bhatt, and R. Raghavan, "The Complexity of Design Automation Problems", Dept. of Computer Science, Univ. of Minnesota, Technical Report 80-23, 1980.

Squeezing Constant Factors of Geometric Algorithms[1]

Jon Louis Bentley
Departments of Computer Science and Mathematics
Carnegie-Mellon University
Pittsburgh, PA 15213

Abstract -- Most theoretical work on geometric complexity has concentrated on the asymptotic efficiency of algorithms. In this paper we will consider improving the constant factors of geometric algorithms. We will first study a particular example in which we achieve a speedup of a factor of fifty, and then consider general principles for realizing such speedups and the implications of the existence of the speedups.

1. Introduction

"Big-oh" analyses of algorithms are dominant in algorithm design for a number of convincing reasons; Aho, Hopcroft and Ullman [1974, Section 1.1] eloquently support this position. Unfortunately, ignoring constant factors can be misleading if one is interested in solving real problems. The purpose of this paper is to demonstrate the importance of this problem and to consider a set of general techniques by which one can efficiently implement asymptotically good geometric algorithms. This effort is for the benefit of two camps: theoreticians can consider asymptotic efficiency without worrying about constant factors (knowing that they can probably be squeezed later), and practitioners can use efficient implementations of asymptotically good algorithms to solve their real problems.

I am currently writing a book about "squeezing constant factors" from programs (Bentley [1982]). There is a great deal of context in that effort that can only be briefly described here, but two important facts must be stated in any discussion of the subject. The most important fact is that techniques for writing efficient code can be easily misapplied to yield incorrect (and often also inefficient systems); these techniques should therefore be applied rarely, and only after certain other actions have been taken. Another important fact is that there are many design levels at which we can achieve efficiency in a software system: the overall system design, selection of algorithms and data structures, translation to machine code, underlying operating system efficiency, and underlying hardware speedups; Reddy and Newell [1977] discuss these design levels and several others. One must therefore be careful to work at the proper design level to avoid being penny-wise and pound-foolish. Nonetheless, and contrary to some commonly-accepted assumptions in the algorithms community, squeezing constant factors is sometimes an important tool in the construction of software systems.

In Section 2 we will study a particular geometric algorithm and see in detail how its Pascal implementation can be speeded up by a factor of six. In Section 3 we will survey how additional

---

[1] This research was supported in part by the Office of Naval Research under Contract N00014-76-C-0370.

techniques yield an additional speedup of over a factor of eight, for a total speedup of a factor of almost fifty. Finally, conclusions about this work are offered in Section 4.

2. An Example

The object of our study in this section will be the following heuristic for the planar Traveling Salesman Problem.

> **The Nearest Neighbor Heuristic:** Choose an arbitrary starting point, and then repeatedly visit the unvisited point closest to the current point until all points have been visited. When this is accomplished, close the tour by returning to the starting point.

Bentley and Saxe [1980] give details on the performance of this heuristic.

The purpose of this exercise is to study the improvements we can achieve by writing efficient code; we will therefore explicitly ignore improvements we could achieve at the higher design level of changing algorithms and at the lower design level of coding with a particular compiler/hardware system in mind. As we study the Pascal programs for the PDP-KL10 we will not examine the machine code that the compiler generates; our only view of the program's speed will be by timing program executions. The Hamburg Pascal compiler used for this experiment is a typical compiler that does little optimization, so the computation that we see in the source code accurately reflects the machine code from which times were collected. Details on the compiler and the timings can be found in Appendix I of Bentley [1982]. To make sure that the timings are not merely artifacts of the particular compiler and hardware, the programs were also implemented in a different language on a different computing system (C on an HP-1000), and produced a similar set of relative timings (those timings can also be found in that Appendix).

The first subroutine for implementing the nearest neighbor heuristic is shown in Fragment A1. It assumes several external definitions: a variable of type `PtPtr` (for "point pointer") is an integer in the range `1..MaxPts`, where `MaxPts` denotes the maximum possible number of points in the plane. The points themselves are stored in an array `PtArr[PtPtr]`, whose elements are records with the two real components `X` and `Y`. The number of points in the array is stored in the integer variable `NumPts`; we will refer to `NumPts` as *N*, since it is the problem size. The subroutine `Dist` is passed two `PtPtr`'s, and returns the Euclidean distance between the two points (the code for the function will be shown in Fragment A3).

The operation of procedure `ApproxTSTour` is straightforward. The only data structure it uses besides the array `PtArr` is an array `Visited[PtPtr]` of `boolean`; the element `Visited[I]` is `true` if and only if point `I` has already been visited in the tour. The routine's first action is to initialize every element of that array to `false` and then choose the first point to be visited as `PtArr[NumPts]` (the last point in the array). It then goes into a loop in which it selects the next `NumPts` – 1 points on the tour. To select each point it finds the closest point not yet visited, and then makes that point the current point. The `writeln` statements produce a description of the tour on the output file; they were not present in the version used for the timings and will not henceforth be shown in the program fragments. The program is simple; excluding lines that contain only `begin`, `end` or `writeln` statements, it contains only thirteen executable lines of code.

```
procedure ApproxTSTour;
     var I, J: PtPtr;
          Visited: array [PtPtr] of boolean;
          ThisPt, ClosePt: PtPtr;
          CloseDist: real;

     begin
     (* Initialize unvisited points *)
     for I := 1 to NumPts do
          Visited[I] := false;

     (* Choose NumPts as starting point *)
     ThisPt := NumPts;
     Visited[NumPts] := true;
     writeln('First city is ', NumPts);

     (* Main loop of nearest neighbor heuristic *)
     for I := 2 to NumPts do
          begin
          (* Find nearest unvisited point to ThisPt *)
          CloseDist := maxreal;
          for J := 1 to NumPts do
               if not Visited[J] then
                    if Dist(ThisPt, J) < CloseDist then
                         begin
                         CloseDist := Dist(ThisPt, J);
                         ClosePt := J
                         end;
          (* Report closest point *)
          writeln('Add edge from ', ThisPt, ' to ', ClosePt);
          Visited[ClosePt] := true;
          ThisPt := ClosePt
          end;

     (* Finish tour by returning to start *)
     writeln('Add edge from ', ThisPt, ' to ', NumPts)
     end;
```

Fragment A1. Original code.

The main `for` loop of the program is executed $N-1$ times, and contains an inner loop that is itself executed N times; the total time required by the program is therefore $O(N^2)$. The Pascal running time of Fragment A1 was observed to be approximately $47.0N^2$ microseconds.

As we modify the program to increase its speed, we will concentrate on the inner loop ("`for J := 1 to NumPts do` ..."), because statements in that loop are executed $N^2 - N$ times, while all other statements are executed at most N times. The first thing that we might notice is that the real-valued result of `Dist(ThisPt,J)` can be calculated twice for each distinct value of J in the inner loop. We will instead calculate it just once, store it, and then use that stored value twice. Since this can cut down the number of distance calculations by a factor of two, we might expect it to cut the run time of the program almost in half. The modified code is shown in Fragment A2;

it stores `Dist(ThisPt,J)` in the real variable `ThisDist`. All the lines that have been changed from Fragment A1 are marked with a vertical bar on their right.

```
    begin                           |
    ThisDist := Dist(ThisPt, J);    |
    if ThisDist < CloseDist then    |
        begin
        CloseDist := ThisDist;      |
        ClosePt := J
        end
    end                             |
```

Fragment A2. Store ThisDist.

When I first made this change I was eagerly waiting to see a factor of almost two squeezed out of the runtime, and I was shocked to see it drop from $47.0N^2$ microseconds to $45.6N^2$ microseconds! After observing these times, though, it was easy to explain what had happened. The `then` clause of the inner `if` statement is executed rarely, so in Fragment A1 the subroutine `Dist` was usually called only once per loop. Specifically, under reasonable probabilistic assumptions, the average number of times that the `then` branch is executed when there are M points left unvisited is the M-th harmonic number, H_M (H_{1000} is about 7.5). For a more detailed analysis of this problem, see Section 1.2.10 of Knuth [1968]. This example illustrates a common experience in writing efficient code: optimizations that we expect to yield a big time savings often make but a small difference. Even though this improvement did not greatly reduce run time, it did identify an important part of the problem: computing the distances between pairs of points.

Is there any way we can improve the distance calculation procedure `Dist` shown at the top of Fragment A3? Unfortunately it appears that we cannot; that procedure succinctly expresses the mathematical definition of Euclidean distances. We can, however, solve a different problem more efficiently: since all we ever do (in this procedure) is compare the relative magnitude of two distances, we do not need to take the square root of the sum of the squares before we return the result. That is, we can compare the squares of the distances to decide which point is closer; this relies on the monotonicity of the square root function. The resulting code is shown at the bottom of Fragment A3.

```
function Dist(I,J: PtPtr): real;
    begin
    return sqrt(sqr(PtArr[I].X-PtArr[J].X) +
                    sqr(PtArr[I].Y-PtArr[J].Y))
    end;

function DistSqrd(I,J: PtPtr): real;                    |
    begin                                               |
    return sqr(PtArr[I].X-PtArr[J].X) +                 |
                    sqr(PtArr[I].Y-PtArr[J].Y)          |
    end;                                                |

...
ThisDist := DistSqrd(ThisPt, J);                        |
...
```

Fragment A3. Remove square roots.

This improvement yields a substantial time savings: while Fragment A2 required $45.6N^2$ microseconds, Fragment A3 requires only $24.2N^2$ microseconds. This difference is almost a factor of two. Since removing $\binom{N}{2}$ square roots saved $21.4N^2$ microseconds, we can deduce that each square root required about 43 microseconds.

There is still one glaring deficiency in the organization of the program. Suppose we are solving the thousand-city problem and we have only ten unvisited cities; how do we find the closest city to `ThisPt`? We look at all one thousand cities, only to find for most of them that they have already been visited. It would be more efficient for us to keep track of the unvisited cities in a more direct way, so we could ignore the visited cities after having visited them. This is accomplished in Fragment A4, which is a complete rewrite of Fragment A1, incorporating the changes of Fragments A2 and A3. (This is the most substantial change we will make to the Pascal program, and indeed the most substantial *kind* of change that falls under the heading of "writing efficient code". This change might be better classified as a selection of data structures, and therefore at a higher design level.) The array `UnVis` contains integer pointers (that is, `PtPtr`'s) to unvisited members of `PtArr`; specifically, the unvisited cities can always be found in `UnVis[1..HighPt]`. The overall structure of the routine is almost unchanged. The initialization is somewhat different, and the structure of the inner loop is different: the `for` statement runs from 1 to the current number of points (`HighPt`), and no `if` test is required. When the closest point has been identified it is swapped with the point in `UnVis[HighPt]`; this maintains the invariant condition that all unvisited cities can be found in `UnVis[1..HighPt]`.

```
procedure ApproxTSTour;
    var
        I: PtPtr;
        UnVis: array [PtPtr] of PtPtr;
        ThisPt, HighPt, ClosePt, J: PtPtr;
        CloseDist, ThisDist: real;

    procedure SwapUnVis(I, J: PtPtr);
        var Temp: PtPtr;
        begin
        Temp := UnVis[I];
        UnVis[I] := UnVis[J];
        UnVis[J] := Temp
        end;

    begin
    (* Initialize unvisited points *)
    for I := 1 to NumPts do
        UnVis[I] := I;

    (* Choose NumPts as starting point *)
    ThisPt := UnVis[NumPts];
    HighPt := NumPts-1;

    (* Main loop of nearest neighbor tour *)
    while HighPt > 0 do
        begin
        (* Find nearest unvisited point to ThisPt *)
        CloseDist := maxreal;
        for I := 1 to HighPt do
            begin
            ThisDist := DistSqrd(UnVis[I],ThisPt);
            if ThisDist < CloseDist then
                begin
                ClosePt := I;
                CloseDist := ThisDist
                end
            end;
        (* Report this point *)
        ThisPt := UnVis[ClosePt];
        SwapUnVis(ClosePt,HighPt);
        HighPt := HighPt-1
        end
    end;
```

Fragment A4. Convert boolean array to pointer array.

The result of this change is to decrease the running time from $24.2N^2$ microseconds to $21.2N^2$ microseconds; it cut the loop overhead in half and eliminated testing, but introduced indirect addressing through the array `UnVis` that was not present before.

From this point on we will concentrate on the inner `for` loop, which is responsible for almost all of the run time. From our knowledge that the `if` test is rarely successful, we can deduce that most of the time is spent in the subroutine `DistSqrd`. To reduce its time, we will rewrite its body in line, which eliminates the overhead of the procedure calls. We then observe that some invariant expressions are reevaluated each time through the loop (namely, the array indexing of

`PtArr[ThisPt]`), so we instead assign those outside the loop to the real variables `ThisX` and `ThisY`. The resulting code is shown in Fragment A5; its running time is $14.0N^2$ microseconds (a reduction of $7.2N^2$ from Fragment A4).

```
(* Find nearest unvisited to ThisPt *)
ThisX := PtArr[ThisPt].X;                        |
ThisY := PtArr[ThisPt].Y;                        |
CloseDist := maxreal;
for I := 1 to HighPt do
    begin
    ThisDist := sqr(PtArr[UnVis[I]].X-ThisX)     |
                + sqr(PtArr[UnVis[I]].Y-ThisY);  |
    if ThisDist < CloseDist then
        begin
        ClosePt := I;
        CloseDist := ThisDist
        end
    end;
```

Fragment A5. Rewrite procedure in line and remove invariants.

We can now see precisely where the time of the program is spent. When M cities are unvisited, it calculates `ThisDist` exactly M times, makes M comparisons with `CloseDist`, and then executes the `then` branch H_M times, on the average. Since the H_M term is too small to worry about (remember, it is 7.5 out of 1000) and all M comparisons seem necessary, we had better concentrate on calculating `ThisDist`. It contains two terms; is there some way we can reduce them to one? Such a reduction is shown in Fragment A6: we first compute the x-distance from the $\mathtt{I}^{\mathrm{th}}$ point to `ThisVert`, and if that alone is greater than `CloseDist`, then we need not examine the y-distance. (Because the second term is positive, it can only increase `ThisDist`.)

```
ThisDist := sqr(PtArr[UnVis[I]].X-ThisX);                        |
if ThisDist < CloseDist then                                     |
    begin                                                        |
    ThisDist := ThisDist + sqr(PtArr[UnVis[I]].Y-ThisY);         |
    if ThisDist < CloseDist then
        begin
        ClosePt := I;
        CloseDist := ThisDist
        end
    end                                                          |
```

Fragment A6. Delay computing y-distance.

Fragment A6 will be faster than Fragment A5 if the x-distance alone is usually sufficient to discard the point from consideration. A heuristic analysis suggests and experimental evidence supports the conjecture that the number of times the y-distance must be considered is only about $2.25M^{1/2}$, where M is the current number of points; thus for 1000 points, only about 70 need have their y-values examined. (Details on the number of y-values examined can be found in Appendix I of Bentley [1982].) The empirically observed running time of Fragment A6 confirms the efficacy of the approach: it reduced the running time from the $14.0N^2$ microseconds of Fragment A5 to $8.2N^2$ microseconds.

3. Further Improvements

In the previous section we improved the speed of a particular Pascal by a factor of almost six. In this section we will survey how that program can be improved even further; details on these improvements can be found in Chapter 2 and Appendices I and II of Bentley [1982].

The next improvements to the program can be realized in the Pascal language. Fragment A7 of Bentley [1982, Chapter 2] achieves a run time of $7.5N^2$ microseconds by converting all arithmetic from integer to real; this means that the algorithm produces only a controlled approximation to the (already approximate) nearest neighbor tour. Fragment A8 reduces the run time to $6.9N^2$ microseconds by using an array created in the previous transform to remove a level of indirect addressing. Finally, Fragment A9 yields a run time of $6.8N^2$ microseconds by removing the cost of index testing by use of sentinels.

The improvements to the run time of the Pascal program are summarized in Table 1. That table describes both the fragments themselves and the techniques used to transform each fragment to the next. The techniques are given as both a brief description of the process and as one of the general rules described in Bentley [1982, Chapters 4 and 5].

Fragment	Modification	General Rule	Time/N^2	Time Change
A1.			47.0	
	Store ThisDist	Expression Rule 3 -- Common Subexpression Elimination		1.4
A2.			45.6	
	Remove square roots	Expression Rule 2 -- Exploit Algebraic Identities		21.4
A3.			24.2	
	Convert boolean array to pointer array	Space-For-Time Rule 1 -- Data Structure Augmentation		3.0
A4.			21.2	
	Rewrite procedure inline and remove invariants	Procedure Rule 1 -- Collapsing Procedure Hierarchies Loop Rule 1 -- Code Motion Out Of Loops		7.2
A5.			14.0	
	Delay computing y-distance	Logic Rule 2 -- Short-circuiting Monotone Functions		5.8
A6.			8.2	
	Convert reals to integers	Expression Rule 2 -- Exploit Algebraic Identities		0.7
A7.			7.5	
	Remove unvisited array	Space-For-Time Rule 1 -- Data Structure Augmentation		0.6
A8.			6.9	
	Put loop control inside inner test	Loop Rule 2 -- Combining Tests (Sentinels)		0.1
A9.			6.8	

Table 1: Summary of program improvements.

Bentley [1982, Chapter 2 and Appendix I] goes on to consider further improvements to the program that can be achieved by coding the inner loop in assembly language. At this point the underlying system is changed from observed run times under the Hamburg Pascal compiler on a PDP-10 to analytic times under an IBM Pascal compiler on the IBM System/360 with the cost model described by Knuth [1971]. Careful (but not clever) translation of Fragment A9 into assembly code results in a program that is over seventeen times faster than the code the compiler produced from Fragment A1. In that speedup, a factor of about 7.0 is accounted for by the Pascal transformations and a factor of about 2.5 is due to careful hand-coding. Applying other techniques leads to an additional speedup of over a factor of three, for a total speedup of a factor of approximately fifty. Some of those additional techniques are quite general (including loop unrolling and an algebraically identical but computationally more efficient representation of one-dimensional ranges), while others are particular to the System/360 architecture (particularly, a clever use of the powerful `BXH` and `BXLE` instructions suggested by Professor G. L. Steele, Jr.).

4. Conclusions

In Section 2 we studied in detail the process of speeding up a Pascal implementation of the Nearest Neighbor Heuristic for the Traveling Salesman Problem; we were able to increase the speed of a program by a factor of over six by applying a set of general techniques. In Section 3 we then briefly surveyed how further techniques could be used to realize an additional speedup of a factor of eight. The techniques used to achieve an overall speedup of a factor of fifty can be roughly classified as follows.

Modification	Speedup Factor
Pascal Improvements	
Geometry-specific (Remove square roots)	2.0
General techniques	3.5
Assembly Language Improvements	
Careful hand coding	2.5
Geometry-specific (Representation of ranges)	3.0

This example provides an instance of the "big-oh" notation ignoring a constant factor of fifty. Factors this large are not uncommon in such endeavors; Bentley [1982] describes several examples of such speedups. This situation can, of course, have a substantial impact on implementations. Unfortunately, even this extreme problem is further exacerbated by the fact that big-oh's of other analyses might hide additional factors that are almost as large. For instance, Bentley, Weide and Yao [1980] describe a linear-time algorithm for a geometric task, and compare it to a straightforward quadratic algorithm. The quadratic program has a run time of $52N^2$ microseconds, while the linear-time program has a run time of $2765N$ microseconds; thus a big-oh analysis of the two algorithms hides a factor of fifty. If the techniques of the previous two sections were applied to the quadratic algorithm, then they might also increase its speed by a factor of fifty, thereby reducing its run time to approximately $1.0N^2$ microseconds.

(This is only a conjecture, and not an experimental result.) If that conjecture is true, then it implies that the original big-oh analysis hides a factor of 2500.

I would like to end this paper by briefly considering what the above facts and conjectures imply about the relevancy of the currently dominant paradigm of algorithm analysis. Algorithm analysts have always known that there are constant factors hidden in our big-ohs; that is precisely the reason we employ the notation. If our goal is proving certain kinds of theorems, then ignoring constant factors does no harm. Even if our goal is implementing an algorithm as a component in a software system, then ignoring constant factors can sometimes be profitable: it allows us to focus our attention on critical parts of the system. If the constant factors we ignore are small (say, less than five), then most programmers can use *ad hoc* techniques to avoid any disastrous effects. Constant factors can, however, be substantially larger than five. A concrete example in this paper showed a constant factor of fifty, and a strongly-supported conjecture would imply a constant factor of *twenty-five hundred*. If these numbers are typical (and Bentley [1982] describes several other examples of similar figures), then they naturally raise the following questions.

> *What is the relevance of the "big-oh" paradigm of algorithm analysis to the task of constructing software systems if those analyses hide constant factors of twenty-five hundred? Is relevance to software systems an important property? And depending on the answers to the above questions, is it time to question the dominant paradigm of algorithm design?*

References

Aho, A. V., J. E. Hopcroft and J. D. Ullman [1974]. *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA.

Bentley, J. L. [1982]. *Writing Efficient Programs*, book in preparation.

Bentley, J. L. and J. B. Saxe [1980]. "An analysis of two heuristics for the euclidean travelling salesman problem," *Eighteenth Annual Allerton Conference on Communication, Control and Computing*, October 1980, pp. 41-49.

Bentley, J. L., B. W. Weide and A. C. Yao [1980]. "Optimal expected-time algorithms for closest-point problems," *ACM Transactions on Mathematical Software* 6, 4, December 1980, pp. 563-580.

Knuth, D. E. [1971]. "An empirical study of FORTRAN programs," *Software--Practice and Experience* 1, 2, pp. 105-133.

Knuth, D. E. [1973]. *The Art of Computer Programming, volume 3: Sorting and Searching*, Addison-Wesley, Reading, Massachusetts.

Reddy, R. and A. Newell [1977]. "Multiplicative speedup of systems," in *Perspectives on Computer Science*, A. K. Jones (ed.), pp. 183-198. Academic Press, New York, NY.

A LINEAR-TIME ALGORITHM FOR DECOMPOSING RECTILINEAR STAR-SHAPED POLYGONS INTO CONVEX QUADRILATERALS

Jörg-R. Sack
Godfried T. Toussaint
School of Computer Science, McGill University
Montreal, Quebec
Canada H3A 2K6

ABSTRACT

Frequently, decomposition techniques have been used to decompose polygons into simpler components in order to analyse the shape of objects. Recently it has been shown that, whereas arbitrary simple polygons do not necessarily admit a decomposition into convex quadrilaterals without using new vertices, such decompositions always exist if the polygons are rectilinear. In this paper an O(n), and thus optimal, algorithm is presented for decomposing rectilinear star-shaped n-vertex polygons into convex quadrilaterals without creating new vertices.

1.INTRODUCTION

The decomposition of simple polygons into simpler components plays an important role in syntactic pattern recognition [4, 9]. Often decompositions into convex or star-shaped parts are of interest [1, 2, 10]. In some applications one might want to decompose objects into fixed "shape primitives," like triangles or quadrilaterals. The term "shape primitives" was introduced by Pavlidis [9]. For a survey of polygon decomposition methods see [14].

Definition: A rectilinear polygon $P = p_1, p_2, \ldots, p_n$ is a simple polygon in which all the edges are parallel to two given orthogonal directions. Without loss of generality, in this paper we consider the x and y axes as the two directions since an arbitrary n-vertex polygon can be rotated in O(n) time. The vertices are given in clockwise order.

Definition: A polygon P is star-shaped if there exists at least one point x inside P such that x can "see" the entire polygon. We say that x can "see" the entire polygon if for every y in P the line segment xy lies entirely in P.

Definition: The set of all points x in P such that x can see the entire polygon is called the kernel of P.

Definition: A polygon P is weakly visible from an edge pq if for every point x in P there exists a point y in pq that sees x.

Definition: P is edge-visible if there exists at least one edge from which the polygon is weakly visible.

Definition: The y-coordinate of a point p will be denoted by y(p).

In image processing rectilinear polygons are frequently used to efficiently describe images on rectangular grids [5, 12]. In this paper we consider convex quadrilaterals as shape primitives and rectilinear star-shaped polygons (RSPs) as objects. We want to decompose rectilinear star-shaped polygons into convex quadrilaterals efficiently. An O(n log n) algorithm for decomposing a polygon into star-shaped polygons has been presented by D. Avis and G.T. Toussaint [1]. An arbitrary simple polygon always admits a triangulation and an O(n log n) algorithm for triangulating a simple polygon is given in [6]. Star-shaped polygons can be triangulated in O(n) time [3, 11]. Kahn, Klawe, and Kleitman [7], in what they refer to as quadrilaterization, proved that it is always possible to decompose a rectilinear polygon into convex quadrilaterals without creating new vertices. This is not true of arbitrary simple polygons. However, they do not present an algorithm and their proof does not suggest an efficient algorithm.

In this paper an O(n) and thus optimal algorithm is presented for decomposing rectilinear star-shaped polygons into convex quadrilaterals. The obvious way of doing this, namely to triangulate the polygon and then to remove every other edge, fails. The reason for this is that such a decomposition may contain concave quadrilaterals, as illustrated in Fig. 1. A correct decomposition of the polygon in Fig. 1 is illustrated in Fig. 2. Our approach is to split the polygon P into two edge-visible polygons, then to decompose each of these polygons, and finally to merge the two decompositions.

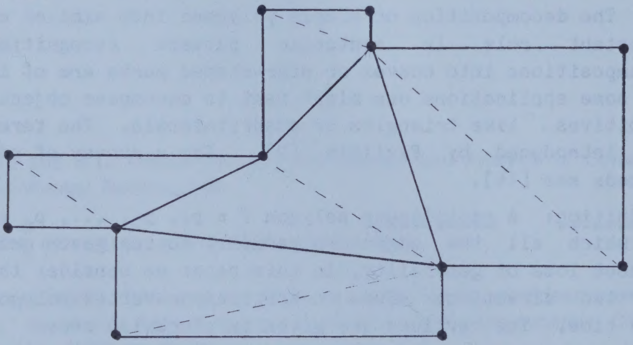


Fig.1

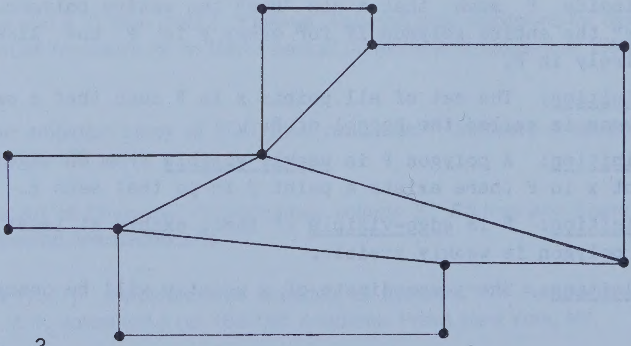


Fig. 2

2. DECOMPOSING RSPs INTO PYRAMIDS

First we describe some properties of rectilinear star-shaped polygons and introduce some notation. It is easy to prove that if the rectilinear polygon is star-shaped then there are exactly two edges with extreme y-coordinates and two edges with extreme x-coordinates. Denote the vertices of the extreme edges as in Fig. 3.

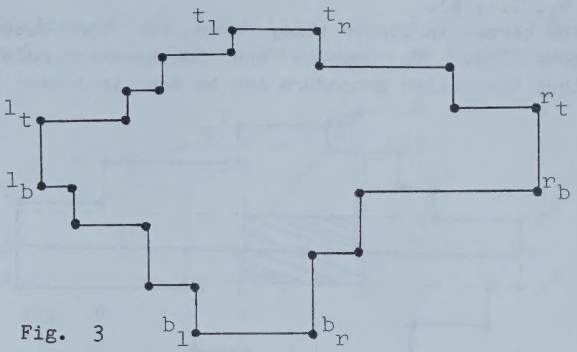


Fig. 3

Definition: A stair is a rectilinear polygonal chain monotone in both the x and y directions.

Lemma: The extreme edges of an RSP are connected by stairs.
The proof is straight forward and left as an exercise to the reader.

Definition: A vertex v is a reflex vertex, if the interior angle at v is greater than 180°; otherwise v is a non-reflex vertex.

If we extend the horizontal and vertical edges at a reflex vertex towards the interior of the polygon, we get the shaded area in Fig. 4. Schachter [10] calls this area the inner cone.

Definition: A reflex vertex v is up-looking, if the inner cone is not below the horizontal line through v; else v is down-looking. In Fig. 4 (a) v is an up-looking reflex vertex, whereas in (b) v is a down-looking reflex vertex.

Definition: A pyramid is defined as a rectilinear edge-visible star-shaped polygon with two stairs.

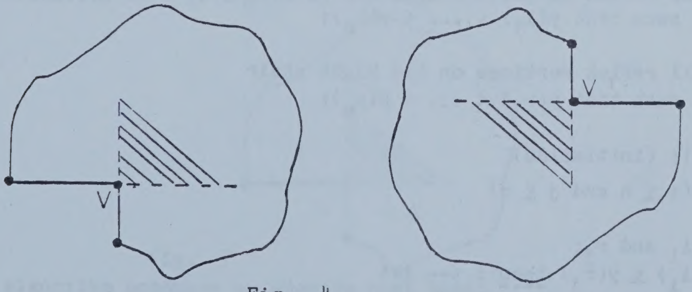


Fig. 4

(a) (b)

Now we show how to decompose an RSP into two pyramids. First find the kernel; this is straightforward once the extreme points are known. The kernel contains at least one point x. Then we construct a horizontal line through x. This line intersects the polygon P at two points say p and q. Because P is star-shaped p and q are located on the x_{min}, x_{max} edges. With these two "Steiner" points p and q we can split the polygon into two pyramids (see Fig. 5). The two pyramids are $(p, \ldots, t_l, t_r, \ldots, q)$ and $(q, \ldots, b_r, b_l, \ldots, p)$.
We can find the kernel in linear time, even for the case of arbitrary simple polygons [8]. To compute the intersection points p, q is then trivial, so that the entire procedure can be done in linear time.

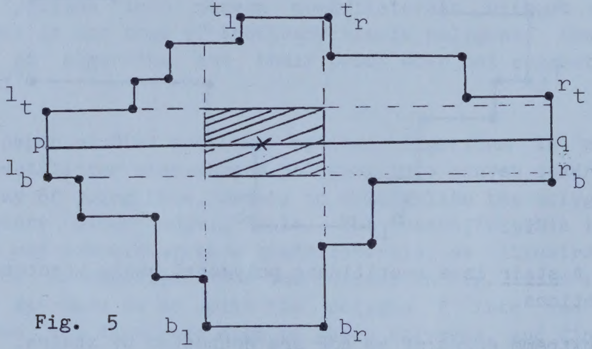


Fig. 5

3. DECOMPOSING PYRAMIDS INTO CONVEX QUADRILATERALS

In this section we show how to decompose a pyramid into convex quadrilaterals. Without loss of generality we assume that the pyramid is oriented in such a manner that $\overline{pq}$ is the top edge. Let $L = l_1, \ldots, l_n$ and $R = r_1, \ldots, r_m$ be two lists of vertices with n, m elements respectively.

Input: A pyramid P.
Output: A convex quadrilaterization of P.

```
Algorithm I
Step 1:
    Starting at the $y_{min}$ edge $\overline{rs}$ construct two lists L, R:
    L <-- all reflex vertices on the left stair
          such that $y(l_1) < \ldots < y(l_n)$;

    R <-- All reflex vertices on the right stair
          such that $y(r_1) < \ldots < y(r_m)$;
Step 2:
    i=1; j=1; (initialize)
    While (i $\leq$ n and j $\leq$ m)
    begin
      join $l_i$ and $r_j$;
      if $y(l_i) \leq y(r_j)$ then i <-- i+1
      else j <-- j+1;
    end
```

Step 3:

```
    If (i > n) join all points $r_j$, ..., $r_m$ to p;
    If (j > m) join all points $l_i$, ..., $l_n$ to q;
```

End of algorithm I

Remark:

In the algorithm, if $y(r_j) < y(l_i) < y(l_{i+1}) < \dots < y(l_k) \leq y(r_{j+1})$ then all points $l_i, \dots, l_k$ are joined to r_{j+1}, as illustrated in Fig. 6.

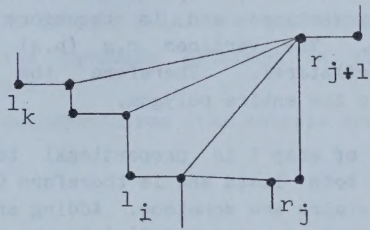


Fig. 6

Lemma 1: An n-vertex pyramid can be decomposed into convex quadrilaterals in O(n) time.

Proof: If both stairs are empty, the polygon is already a convex quadrilateral and the algorithm stops. Let the polygon have at least one non-empty stair. The algorithm creates polygons, say $Q_1, \dots, Q_d$. We have to show that:

(i) $Q_1, \dots, Q_d$ are quadrilaterals.
(ii) $Q_1, \dots, Q_d$ are convex.
(iii) No two polygons Q_i, Q_j intersect (properly).
(iv) The union of all $Q_1, \dots, Q_d$ is equal to the polygon.
(v) The algorithm runs in linear time.

(i) From the algorithm it is clear that every pair of consecutive reflex vertices in the left (right) stair is joined to a common vertex in the right (left) stair, therefore the components are quadrilaterals.

(ii) Let v be a reflex vertex on the left stair and refer to Fig. 7. The concavity at v is broken by any edge located in region B.

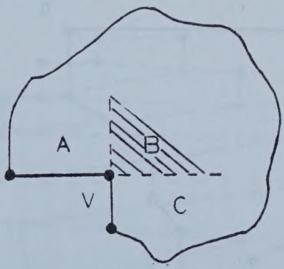


Fig. 7

The algorithm produces an edge vw such that:

(a) w is located on the other stair; therefore $\overline{vw}$ is not located in region A.

(b) $y(v) \leq y(w)$; therefore vw is not located in region C. All other reflex vertices are handled in a similar way. Since all other angles in the quadrilaterals are either 90° or less, the quadrilaterals are convex.

(iii) Since the vertices are scanned in order by increasing y-coordinates, it follows trivially that no two quadrilaterals intersect.

(iv) Each reflex vertex is contained in at least two quadrilaterals. Each non-reflex vertex (except r, s (p, q) the bottom (top) vertices) is located between two reflex vertices and is therefore contained in the respective quadrilateral. The vertices r,s (p,q) are included in the lowest (highest) quadrilateral. Therefore the union over all quadrilaterals is equal to the entire polygon.

(v) The running-time of step 1 is proportional to the sum of the number of elements in both lists and is therefore O(n). For step 1 no sorting is needed as the stairs are monotone. Adding an edge in step 2 and 3 takes constant time and is executed O(n) times. Therefore the total run-time is O(n). Q.E.D.

Remark 1:

Let l be the vertex (on the left stair) joined to p and let r be the vertex (on the right stair) joined to q. If $y(l) > y(r)$ the algorithm joins reflex vertices to q. Otherwise $y(l) \leq y(r)$ and the algorithm joins reflex vertices to p. This follows immediately from the algorithm.

Remark 2:

We now describe a generalization of a pyramid which will be useful in the next section for merging decomposed pyramids. Let P be a pyramid with pq and sr as top and bottom edges, respectively. Let LS(r,p) and RS(q,s) denote the left and right stairs, respectively, of P. In a pyramid any horizontal edge $p_i p_j$ is such that $y(p_i) = y(p_j)$. We can generalize these pyramids to non-rectilinear pyramids by allowing $y(p_i) \leq y(p_j)$ in LS(r,p) and letting $y(p_j) \leq y(p_i)$ in RS(q,s) for any horizontal edge $p_i p_j$ in P. We shall call these polygons worn-out pyramids (see Fig. 8). It is easy to see that with algorithm I these worn-out pyramids can also be decomposed into convex quadrilaterals since the interior angles that were previously 90° are still less than 180° and are therefore non-reflex.

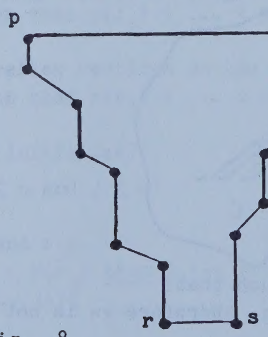


Fig. 8

4. DECOMPOSITION OF RECTILINEAR STAR-SHAPED POLYGONS

In this section we present an algorithm that solves our original problem of decomposing a rectilinear star-shaped polygon P into convex quadrilaterals.

Input: A rectilinear star-shaped polygon P.

Output: A convex quadrilaterization of P.

Algorithm II
Step 1: Decompose the polygon P into two pyramids.

Step 2: Decompose the pyramids obtained in step 1 into convex quadrilaterals.

Step 3: Merge the decompositions (the details are given below).

End of algorithm II

Theorem:
Algorithm II decomposes a rectilinear n-vertex star-shaped polygon into convex quadrilaterals in O(n) time.

Proof:
In section II it was shown that step 1 can be done in O(n) time.

In step 2 we apply algorithm I of section III to both polygons resulting from step 1. In lemma 1 we showed that this can be done in linear time.

In step 3 we have to merge the two decompositions obtained in step 2. As the polygon P is star-shaped there are only two cases to be considered:

(a) The intervals $[y(l_t), y(l_b)]$ and $[y(r_t), y(r_b)]$ are such that neither is contained in the other.

(b) One of the intervals $[y(l_t), y(l_b)]$, $[y(r_t), y(r_b)]$ is contained in the other.

(a) Without loss of generality let $y(l_t) > y(r_t)$ and $y(l_b) > y(r_b)$, and refer to Fig. 9. According to remark 1 in section III we notice that the top polygon-decomposition contains edges from p_i to p for some values of i. Similarly the bottom polygon-decomposition contains edges from q_j to q for some j. Now replace all edges p_ip in the top polygon by p_il_b and all edges q_jq in the bottom polygon by q_jr_t.

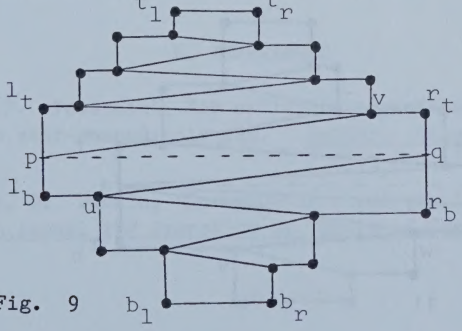


Fig. 9

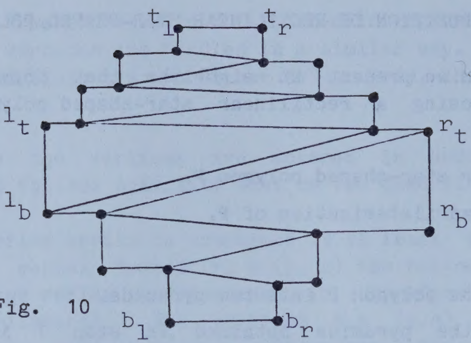


Fig. 10

By doing this we replace in each of the concerned quadrilaterals, located above and below $\overline{pq}$, exactly one vertex, thus obtaining a new quadrilateral. As l_b is below p and r_t is above q, these quadrilaterals remain convex and cannot intersect. Furthermore, the hexagon formed by deleting $\overline{pq}$, namely (p, v, r_t, q, u, l_b), clearly becomes a convex quadrilateral, since it contains two parallel edges $\overline{l_bu}$ and $\overline{vr_t}$. This procedure applied to the polygon of Fig. 9 yields the decomposition of Fig. 10.

(b) Without loss of generality let $[y(l_t), y(l_b)]$ be contained in $[y(r_t), y(r_b)]$ and refer to Fig. 11. In this case $y(l_t) \leq y(r_t)$ and $y(l_b) > y(r_b)$, and with remark 1 in section III we see that in each pyramid one or more reflex vertices are joined to q.

Top polygon

Let t be the highest vertex joined to q.
Let u be the lowest vertex joined to t. $(u \neq q)$

Bottom polygon

Let w be the lowest vertex joined to q.
Let v be the highest vertex joined to w. $(v \neq q)$

Delete all decomposition-edges to q. Furthermore delete p, q and $\overline{pq}$. Then consider the polygon $P^*=(u, r_t, r_b, v, w, \ldots, t)$. As v is not located above w, and u is not located below t, P^* is a worn-out pyramid. Rotate P^* by 270° (clockwise) and decompose it as in the remark 2 in section III. (In practice one would implement algorithm I so that the rotation can be avoided. The bottom-to-top approach of algorithm I would become a left-to-right (or right-to-left) approach.) The new decomposition edges inside P^* cannot interfere with the decompositions outside P^*. Applying this procedure to the decomposition of Fig. 11 yields that shown in Fig. 12.

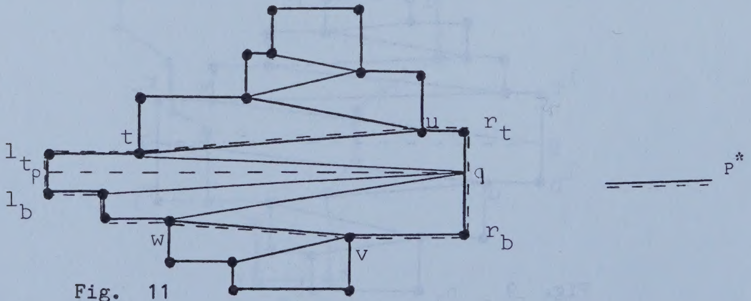


Fig. 11

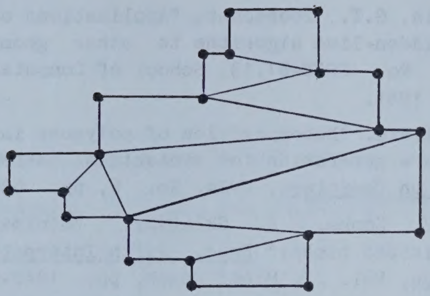


Fig. 12

The merging step is, therefore, either straightforward or applies lemma 1 to P^*. In both cases the run-time is O(n). We can conclude therefore that the total time required to decompose a rectilinear star-shaped polygon into convex quadrilaterals is O(n). Q.E.D.

Remark:

The theorem provides yet another method to triangulate rectilinear star-shaped polygons.

5. CONCLUSIIONS

We have shown how a special class of rectilinear polygons, namely *star-shaped* rectilinear polygons, can be decomposed into convex quadrilaterals. The algorithm we presented, exhibits a run-time proportional to the number of vertices and is thus optimal to within a multiplicative constant. The decomposition of rectilinear star-shaped polygons gives us a good insight into the decomposition problem. The decomposition problem for other classes of rectilinear polygons (including arbitrary rectilinear simple polygons) is under investigation.

One open problem we have not considered is finding the *minimum-weight-quadrilaterization* of a rectilinear star-shaped polygon, where the weight is defined as the sum of the euclidean lengths of the added diagonals.

6. REFERENCES

[1] D. Avis, G.T. Toussaint, "An efficient algorithm for decomposing a polygon into star-shaped polygons," *Pattern Recognition*, in press, 1981.

[2] B. Chazelle, D. Dobkin, "Decomposing a polygon into its convex parts," *11th annual ACM Symposium on Theory of Computing*, pp. 38-48, 1979.

[3] H. ElGindy, D.Avis, G.T. Toussaint, "Applications of a two-dimensional hidden-line algorithm to other geometric problems," Technical Report No. SOCS-81.13, School of Computer Science, McGill University, April 1981.

[4] H. Feng, T. Pavlidis, "Decomposition of polygons into simpler components: feature generation for syntactical pattern recognition," IEEE Transactions on Computers, C-24, No. 6, pp. 636-650, June 1975.

[5] L. Ferrari, P.V. Sonka, J. Sklansky, "Minimal rectangular partitions of digitized blobs," Proc. Fifth International Conf. on Pattern Recognition, Vol. 2, Miami Beach, pp. 1040-1043, Dec. 1980.

[6] M.R. Garey, D.S. Johnson, F.P. Preparata, and R.E. Tarjan, "Triangulating a simple polygon," Information Processing Letters, Vol. 7, pp. 175-179, June 1978.

[7] J.Kahn, M.Klawe, D. Kleitman, "Traditional galleries require fewer watchmen," RJ3021 (37386) 12/22/80, Computer Science Department, IBM, San Jose, California 95193, Mathematics Department, Massachusetts Institute of Technology.

[8] D.T.Lee, F.P.Preparata, "An optimal algorithm for finding the kernel of a polygon," Journal of the ACM, Vol. 26, pp. 415-421, 1979.

[9] T. Pavlidis, "Structural pattern recognition," Springer Series in Electrophysics Vol. 1, Springer Verlag Berlin, Heidelberg, New York, 1977.

[10] B. Schachter, "Decomposition of polygons into convex sets," IEEE Transactions on Computers, C-27, No. 11, pp. 1078-1082, Nov. 1978.

[11] A.A.Schoone, J. van Leeuwen, "Triangulating a star-shaped polygon," University of Utrecht, RUU-CS-80-3, April 1980.

[12] J. Sklansky, "Recognition of convex blobs," Pattern Recognition, Vol. 2, pp. 3-10, 1970.

[13] G.T. Toussaint, D. Avis, "On a convex hull algorithm for polygons and its application to triangulation problems," Pattern Recognition, in press, 1981.

[14] G.T. Toussaint, "Pattern recognition and geometrical complexity," Proc. Fifth International Conf. on Pattern Recognition, Vol. 2, Miami Beach, pp.1324-1347, Dec. 1980.

THE P-CENTER PROBLEM IN THE PLANE IS NP-COMPLETE

A. Marchetti-Spaccamela, IASI-CNR, Via Buonarroti 12, Roma, Italy

ABSTRACT

We consider the problem of choosing k centers among n points in the euclidean plan so that the maximum distance from each of the n points to the closest center is minimized. This problem is shown to be NP-complete.

INTRODUCTION

A well known location problem is considered. Given a graph and a number k, k centers must be chosen among the vertices in such a way that if dist(v) is the length of the shortest path from a vertex v to its closest center then max d(v) is minimized.

This problem is known as the p-center problem and has been shown to be NP-complete in the case of general graphs (see 2).

In this paper the euclidean version of this problem, when the vertices are points of the plane and the distance between any pair of points is the euclidean distance, is shown to be NP-complete (as conjectured by Papadimitriou (4)).

In order to show that the problem is NP-complete we must formulate it as a recognition problem:

Given a multiset of n points $A = \{P_1, \ldots, P_n\}$ and 2 numbers k and d, does there exist a subset $B \subseteq A$ such that $\max_{j \in A} \min_{P_i \in B} \mathrm{dist}(P_i, j) \leqslant d$?

MAIN RESULT

Theorem

The p-center problem in the plane is NP-complete.

Proof

It is trivial to see that the problem is in NP. To prove its NP-hardness we reduce the exact 3 cover problem to the p-center problem.

Exact 3 cover

Given a set $S = \{1,2,\ldots,n\}$, n multiple of 3, and a family $F = \{S_1,S_2,\ldots,S_m\}$ of subsets of S, with $|S_i| = 3$ for all i, does there exist a cover $C \subseteq F$ such that $|C| = \frac{n}{3}$ and $\bigcup_{j \in C} S_j = S$?

This NP-complete problem (see 2) has been used for the reductions given for other geometric problems (1,3,4). Given an instance of exact 3 cover, an instance of the p-center problem with $30\ m \cdot n + 18\ n + 3\ m + 16\ m \cdot n/3 + 14\ n/3$ points and $k = 2(m+1)n/3 + n(4m+1)$ centers is constructed. It needs to be shown whether there is an arrangement of the centers such that every point is covered by at least one circle with ray $d = \sqrt{5}$.

The location of the points is schematically shown in Fig. 1. There are two blocks 1 and 2.

In block 1 there are m+1 S-configurations $S_1,S_2,\ldots,S_{m+1}$ and m T-configurations $T_1,T_2,\ldots,T_m$.

S_i and T_i are divided in n/3 columns S_{ih} and T_{ih}, $h = 1,2,\ldots,n/3$, and are formed by 14 and 2 points respectively. Fig. 2 shows the location of the points.

In block 2 there are m+1 R-configurations $R_1,R_2,\ldots,R_{m+1}$ and m Q-configurations $Q_1,Q_2,\ldots,Q_m$.

Each R_i can be divided in n columns R_{ij}, $j = 1,2,\ldots,n$, each formed by 18 points on 3 parallel lines, 6 for each line. The distance between successive points in each line is 2; the distance between the first and the second line is 0.1 and the distance between the second and the third line is 0.9 (see Fig. 3). Analogously, each Q_i can be divided in n columns Q_{ij}, $j = 1,2,\ldots,n$, such that each Q_{ij} is between R_{ij} and $R_{(i+1)j}$ and has two different configurations depending whether it is $j \in S_i$ or $j \notin S_i$. In this way, the particular structure of the instance of exact 3 cover problem is taken into account.

The following notation will be used to distinguish points of R_{ij}, Q_{ij}, S_{ij} and T_{ij}. With references to Figures 2, 3 point a_k, $k=1,2,\ldots,6$ belonging to R_{ij} will be written $a_k(i,j)$; the same will be done for the other points b_k, c_k, $k=1,2,\ldots,6$, belonging to R_{ij}. A similar notation is used for the points of Q_{ij}, S_{ij} and T_{ij}.

Suppose that there is a feasible solution B for the p-center problem. It will be shown that there is a feasible solution for the exact 3-cover problem.

Proposition 1

If there is a feasible solution B there exists a feasible solution B' which has no centers on points $\ell_1(i,j)$, $\ell_2(i,j)$ of Q_{ij} and points $y_1(i,h)$, $y_2(i,h)$ of S_{ih} for all i, j and h.

Proof

A center placed on $\ell_1(i,j)$ covers $d_8(i,j)$ and 2 other points $a_2(i+1,j)$, $a_3(i+1,j)$. Since every center covering $b_2(i+1,j)$ [$b_3(i+1,j)$] also covers $a_2(i+1,j)$ [$a_3(i+1,j)$] it's possible to move the center on $d_8(i,j)$.

The same idea can be applied to move centers placed on $\ell_2(i,j)$, $y_1(i,h)$, $y_2(i,h)$.■

In the following we will consider the solution B' obtained by applying Proposition 1.

It's easy to see that on each S_{ih} and T_{ih} there is at least one center; moreover, each center can cover 1 point chosen among $y_1(i,h)$, $y_2(i,h)$ $i=1,2,\ldots,m+1$, $h=1,2,\ldots,n/3$ at most. Therefore, the following lemma has been proved.

Lemma 1

There are at least $2(m+1)n/3$ centers placed on points belonging to block 1.

Figure 4 shows how 2(m+1)n/3 centers can cover all the points of block 1 and n/3 tuple $a_1(i,1)$, $b_1(i,1)$, $c_1(i,1)$ for n/3 values of i.

Note that n/3 is the maximum possible number using only 2(m+1)n/3 centers. In fact, for each index i such that a point of R_{i1} has its closest center in block 1, there are at least n/3+1 centers on S_i; moreover each remaining S_i and each T_i require at least n/3 centers. Hence, if there are $v > n/3$ indexes i with the above property, the total number of centers in block 1 is $v(n/3+1) + n(m-v+1)/3 + nm/3 = 2\ nm/3 + n/3 + v > 2n(m+1)/3$. Let us consider the minimum number of centers necessary in block 2.

Proposition 2

On each R_{ij} there are at least 2 centers.

Proposition 3

To cover with centers placed on $R_{ij} \cup Q_{ij}$ all the points that must have their center on R_{ij} or Q_{ij} plus 1,2,3,4 points chosen among $\ell_1(i-1,j)$, $\ell_2(i-1,j)$, $\ell_1(i,j)$, $\ell_2(i,j)$ 4,4,5,6 centers are necessary and sufficient.

Proof

Part 1: To cover $\ell_1(i,j)$ we need 4 different centers for $\ell_1(i,j)$, $a_2(i,j)$, $d_9(i,j)$, $a_5(i,j)$; the other cases are similar.

Part 2: Necessity is implied by Part 1. Figures 5a, 5b show two possible arrangements of 4 centers that cover 2 other points.

Part 3: In order to cover $\ell_1(i-1,j)$, $\ell_2(i-1,j)$, $\ell_1(i,j)$ we need a center on d_8 and two centers on the upper line of R_{ij}. Hence points d_3 and d_9 need to other centers. The other cases are similar.

Part 4: Two centers are necessary for $\ell_1(i,j)$, $\ell_2(i,j)$ on d_8 and d_9; two centers are necessary on the upper line of R_{ij} for $\ell_1(i-1,j)$, $\ell_2(i-1,j)$. So we need another couple for d_3 and d_5.∎

Proposition 4

To cover with centers placed on $R_{ij} \cup Q_{ij}$ all the points that must have their center on R_{ij} or Q_{ij} 4 centers are necessary and sufficient if $j \notin S_i$. If $j \in S_i$ 3 centers are sufficient.

Proof

Part 1: Sufficiency follows from Proposition 3; if $j \notin S_i$ we need two centers on R_{ij} and two on Q_{ij} for d_8 and d_9.

Part 2: Figure 6 shows how 3 centers are sufficient if $j \in S_i$; note that this is the only way to do that and cover 18 points belonging to R_i.

Lemma 2

There are at least $n(4m+1)$ centers in block 2.

Proof

In each column j there are 2m points $\ell_1(i,j)$, $\ell_2(i,j)$. The last pair can be covered by the two centers used for $R_{(m+1)j}$. The remaining $2(m-1)$ pairs need $4(m-1)+3$ centers (by Propositions 3 and 4). As there are n columns the Lemma is proved.∎

Lemmas 1 and 2 show that k, the number of centers available, is the minimum number required in each feasible solution.

Lemma 3

For each i,j there are exactly two centers on R_{ij}. For each i,j on Q_{ij} there are two centers, with the exception of one for each column j.

Proof

Suppose that, for a given $\hat{j}$, for $w(w > 1)$ indexes i there is only one center on $Q_{i\hat{j}}$. Hence there are 2m points that must be covered with centers placed on m-w regions $R_{i\hat{j}} \cup Q_{i\hat{j}}$. In order to do this, $4(m-w) + 2w-2$ centers are necessary (by Proposition 3). So the total number of centers required is $3w + 4(m-w) + 2w - 2 + 2 = 4m+w > 4m+1$. As in the other columns, there are at least 4m+1 centers, this implies that the number of centers used is larger than k. ■

Proposition 5

If there is 1 center placed on $Q_{i\hat{j}}$ then points $a(\hat{i},1)$, $b(\hat{i},1)$, $c(\hat{i},1)$ are not covered by centers placed on block 2.

Proof

By Proposition 4 and Lemma 3 if there is 1 center on $Q_{i\hat{j}}$ then points $a(\hat{i},j)$, $b(\hat{i},j)$, $c(\hat{i},j)$ must have their center in $R_{\hat{i}(j-1)}$. As there are two centers placed on $R_{\hat{i}(j-1)}$ it follows that points $a(\hat{i},j-1)$, $b(\hat{i},j-1)$, $c(\hat{i},j-1)$ must be covered by a center placed on $R_{\hat{i}(j-2)}$. The proposition is proved by applying inductively the same argument. ■

Let us define $G = \{i \mid \text{for some } j \text{ there is exactly one center placed on } Q_{ij}\}$. We will show that G is a solution of the exact 3 cover problem.

Lemma 4

The cardinality of G is n/3.

Proof

By construction and Lemma 3 it is easy to see that the cardinality of G is greater or equal to n/3.

Suppose now that $|G| > n/3$. By Proposition 5 there are $|G|$ triple of points $a_1(i,1)$, $b_1(i,1)$, $c_1(i,1)$ belonging to block 2 that must have their center in block 1. This implies that there are more than 2mn/3 centers in block 1. ■

By combining Lemmas 3 and 4 it is easy to show that a is an exact cover. Now suppose that C is a solution of the exact cover problem. To obtain a solution for the p-center problem it is sufficient to place:

block 1 2n(m+1)/3 centers in such a way to cover also points $a_1(i,1)$, $b_1(i,1)$, $c_1(i,1)$ if $i \in C$;

block 2 one center in Q_{ij} if $i \in C$ and j belongs to S_i, two centers in all other cases; two centers on each R_{ij}.

This completes the proof of the theorem.

References

1. M.R. GAREY, R.L. GRAHAM, D.S. JOHNSON: "Some NP-complete geometric problems" Proc. STOC 1976.

2. M.R. GAREY, D.S. JOHNSON: "Computers and intractability: a guide to the theory of NP-completeness" Freeman, 1978.

3. C.H. PAPADIMITRIOU: "The euclidean travelling salesman is NP-complete" Theor. Comp. Sc. 4 1978.

4. C.H. PAPADIMITRIOU" "Worst case and probabilistic analysis of a geometric problem" Siam. J. on Computing, vol. 10, no. 3, 1981.

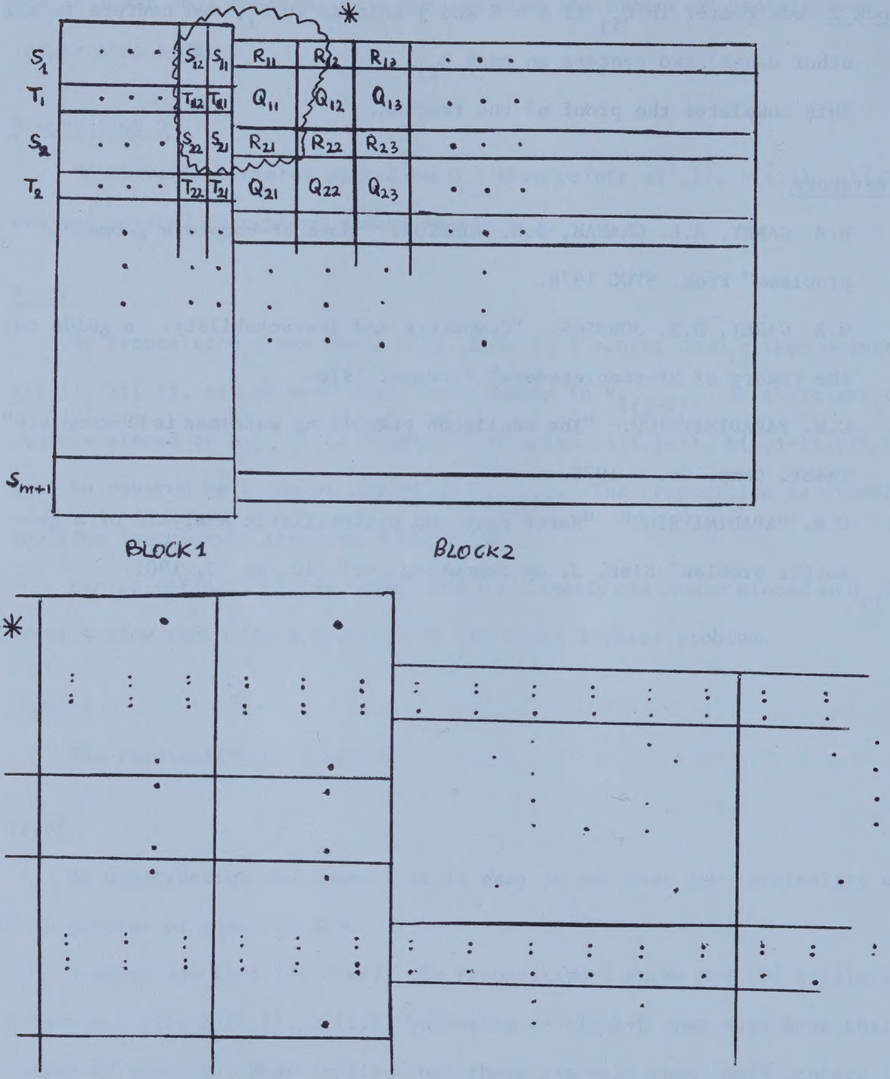
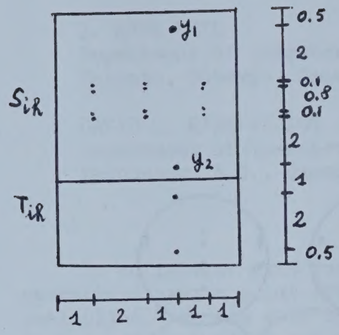
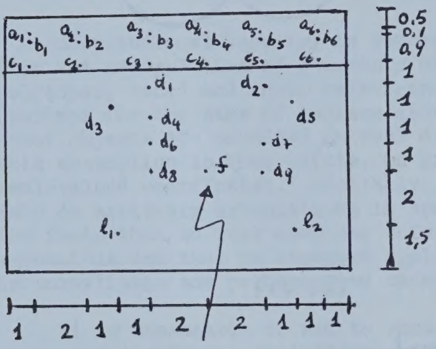

*
S1
T1
S2
T2
Sm+1
S12
S11
T12
T11
S22
S21
T22
T21
R11
R12
R13
Q11
Q12
Q13
R21
R22
R23
Q21
Q22
Q23
BLOCK1
BLOCK2
*

FIG. 1



y_1
y_2
S_{ik}
T_{ik}
0.5
2
0.1
0.8
0.1
2
1
2
0.5
1
2
1
1
1

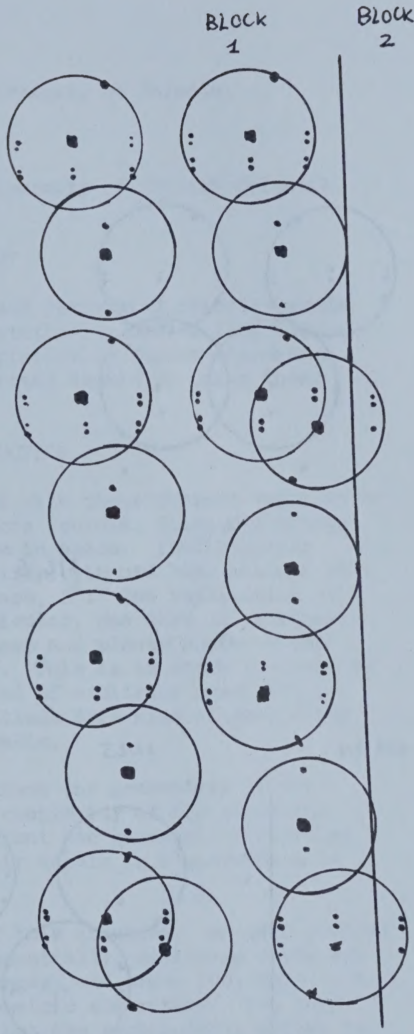
FIG. 2



$a_1 : b_1$ c_1
$a_2 : b_2$ c_2
$a_3 : b_3$ c_3
$a_4 : b_4$ c_4
$a_5 : b_5$ c_5
$a_6 : b_6$ c_6
d_1
d_2
d_3
d_4
d_5
d_6
d_7
d_8
d_9
f
l_1
l_2
0.5
0.1
0.9
1
1
1
1
2
1,5
1
2
1
1
2
2
1
1
1

THERE IS POINT f IFF $x_j \in S_i$

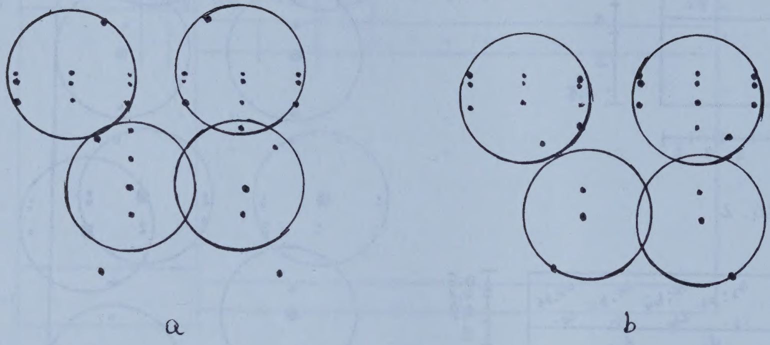
FIG. 3



BLOCK 1
BLOCK 2

POINTS $a_1(2,1)$, $b_1(2,1)$, $c_1(2,1)$
AND $a_1(4,1)$, $b_1(4,1)$, $c_1(4,1)$
ARE COVERED BY A CENTER
PLACED ON BLOCK 1

FIG 4


a

b

FIG. 5

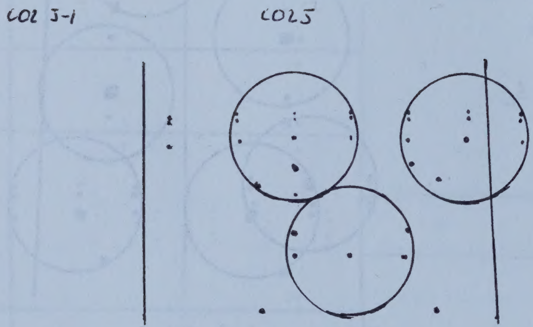

COL J-1
COL J
COL J+1

FIG. 6

COMPUTATIONAL GEOMETRY ON INTEGER GRIDS

J. MARK KEIL
Department of Computer Science, University of Toronto,
Toronto, Ontario, Canada

DAVID G. KIRKPATRICK
Department of Computer Science, University of British Columbia,
Vancouver, B.C., Canada

ABSTRACT

In an integer grid domain, sorting and performing priority queue operations over a point set become asymptotically faster than in the real plane. We are able to design asymptotically faster algorithms for geometric prblems set in this restricted domain by using these faster tools.

1. INTRODUCTION

Computational Geometry is concerned with the efficient representation and manipulation of geometric objects (points, lines, polygons, polytopes, etc.) and their relationships in space. Traditionally (perhaps for the sake of maximum generality), it has been assumed that these objects are embedded in real d-space, R^d. One implication of this assumption is that points, in particular, can take on arbitrary real-valued coordinates. Similarly, lines and planar surfaces can take on arbitrary orientations in space. This is in stark contrast to the facts that on real machines this kind of arbitrary precision is impossible and that in numerous applications this kind of generality is unrealistic and perhaps even undesirable.

It is reasonable to ask to what extent the generality of this unconstrained framework influences the complexity of the resulting algorithms. In other words, to what extent can geometric algorithms be made more efficient by narrowing their domain to a somewhat more restricted object space?

This paper addresses one aspect of this question. We make precise what we call an integer grid domain (essentially, arbitrary reals are replaced by integers from restricted ranges) and argue that this is a realistic setting in which to study geometric algorithms. Two basic algorithm design tools are introduced that are particularly suited to (or exploitive) this domain. A number of applications are described for which asymptotically faster algorithms are possible in this restricted setting.

2. INTEGER GRIDS

An m-grid in R^d is any set of d-tuples (points) each coordinate of which is an integer drawn from some fixed range of size at most m. For simplicity, we will focus our attention on the particular m-grid $\{0,1, \ldots,m-1\}^d$ although in what follows there is nothing to distinguish this from other more general m-grids.

It should be clear that, with appropriate translation and scaling, m-grids can be used to approximate any bounded region of R^d, and if m is chosen sufficiently large this approximation can be arbitrarily dense.

While it is natural to view a k-tuple as a point in R^k, it is also possible, given k unit vectors $v_1,\ldots,v_k$ in R^d, to view the k-tuple $(\lambda_1,\ldots,\lambda_k)$ as denoting the point $\sum \lambda_i v_i$ in R^d. With this more general interpretation, we can define a generalized m-grid in R^d as a set of k unit vectors in R^d together with any set of k-tuples each coordinate of which is an integer drawn from some fixed range of at most m (see Figure 1). Note that the real (Cartesian) coordinates of points on a generalized m-grid may not be rational multiples of one another. Generalized m-grids arise quite naturally in certain applications, in particular in describing the vertices of polyhedra defined by hyperplanes that are orthogonal to and integrally displaced along one of some finite set of normal vectors.

3. SOME TOOLS

In this section we remind the reader of two important algorithm design tools which allow us to exploit the assumption of an integer grid domain. Most of the results are well known, although we do present some new observations on priority queues, specifically a tradeoff between operation-time and space, that should make this tool somewhat more practical in certain applications.

3.1 Integer Sorting

Any collection of $O(n)$ integers in the range $(0,n-1)$ can be sorted in $O(n)$ time using a straightforward bucket sort. This result can be extended, at the expense of additional storage, to the following:

Lemma 3.1 (11) Any collection of n integers in the range $(0,m-1)$ can be sorted in $O(n \log\log_n m + n)$ time using $O(m^{\frac{1}{2}})$ space.

This result can be generalized (see (11)) to use less storage at the expense of a slightly weaker time bound.

As a direct application of lemma 3.1 we can construct efficient algorithms for sorting n points from an m-grid by direction (e.g. lexicographically), by distance or by angle. In particular, if m is any fixed polynomial in n, then this sorting can be done in $O(n)$ time.

Lemma 3.2 Let S be any collection of n points drawn from an m-grid in R^d and let p be any point on the m-grid. Then,

i) S can be sorted in the direction of p in $O(n \log\log_n m + n)$ time;

ii) S can be sorted by distance from p in $O(n \log\log_n m + n)$ time; and

iii) in the case $d = 2$, S can be sorted by angle about p in $O(n \log\log_n m + n)$ time.

Proof. We consider the case $d = 2$ only. The more general results follow by straightforward extensions of our arguments. By lemma 3.1, it suffices to reduce each problem to that of sorting n elements in the range $(0,r-1)$, where r is $O(m^k)$ for some fixed k. Let $S = \{(a_1,b_1),\ldots,(a_n,b_n)\}$ and $p = (x,y)$.

i) To sort S in the direction of p, it suffices to sort the set $S' = \{a_i x + b_i y \mid 1 \leq i \leq n$. Since $0 \leq a_i, b_i, x, y \leq m-1$ every element of S' lies in the range $(0, 2(m-1)^2)$.

ii) To sort S by distance from p, it suffices to sort the set $S' = \{((a_i - x)^2 + (b_i - y)^2)^{\frac{1}{2}} \mid 1 \leq i \leq n\}$ which is equivalent to sorting the set $S'' = \{(a_i - x)^2 + (b_i - y)^2 \mid 1 \leq i \leq n\}$. Since $0 \leq a_i, b_i, x, y \leq m - 1$, every element of S'' lies in the range $(0, 2(m - 1)^2)$.

iii) By appropriate scaling, it is sufficient to consider only the case where $p = (0,0)$. In this case, sorting by angle is equivalent to sorting by slope. That is, the problem is reduced to that of sorting the set $S' = \{ b_i/a_i \mid 1 \leq i \leq n\}$. It is easy to see that any two unequal elements of S' must differ by at least $1/m^2$. Thus sorting S' is equivalent to sorting the set $S^n = \{ \lceil m^2 b_i/a_i \rceil \mid 1 \leq i \leq n\}$ each finite element of which lies in the range $(0,m^3)$. □

3.2 Integer Priority Queues

Any data structure which is capable of maintaining a set with operations of type INSERT, DELETE and MIN has been called a priority queue (1). Many implementations of priority queues also permit efficient execution of the operations PREDECESSOR and SUCCESSOR, and we will consider this part of the specification of a priority queue for the purposes of this paper.

Priority queues using O(logn) operation-time and O(n) space for the maintenance of arbitrary linearly ordered sets of size n are quite well known (1,12). The idea of exploiting a restricted domain to improve the efficiency of priority queue maintenance was first considered by van Emde Boas. His results (6,7) can be summarized as follows:

Lemma 3.3 (7) A priority queue on the integer domain $\{0,\ldots,m - 1\}$ can be maintained in O(m) space and O(loglogm) time per operation, following O(m) preprocessing.

As Johnson (9) points out, the priority queue of van Emde Boas is efficient only for sequences of operations of length n comparable to the domain size m. By showing how much of the pointer structure of van Emde Boas' priority queue can be made implicit (making it possible to avoid or at least defer its initialization) Johnson demonstrates the following:

Lemma 3.4 (9) A priority queue on the integer domain $\{0,\ldots,m - 1\}$ can be maintained in O(m) space and O(loglogm) time per instuction, following only O(1) preprocessing.

A major drawback of Johnson's priority queue (as well as that of van Emde Boas) is its storage requirement. Obviously, a sequence of n priority queue operations can involve only O(n) elements which could be an extremely small fraction of the entire domain. Yet the storage requirement is proportional to the domain size, at least in the worst case. For many applications, notably those we envision in section 4, this requirement is unrealistically large.

However, it is possible to retain the asymptotic improvements in operation times provided by Johnson's priority queue at somewhat less cost in terms of space. In fact there is a tradeoff possible between operation times and space that provides a full spectrum of complexities for priority queues. Specifically,

Theorem 3.5 A priority queue on the integer domain $\{0,\ldots,m - 1\}$ can be maintained in $O(\frac{n\log m}{\log t} + nt)$ space and $O(\log m \log\log t/\log t)$ time per operation, following $O(1)$ preprocessing, for any $t \leq m$.

Proof. The basic idea is to construct a priority queue out of many layers of smaller priority queues equivalent to those described by Johnson (9).

Imagine to start that we have a structure P built out of $k = \log_t m$ levels of priority queues each of which manipulates subsets of $\{0,\ldots,t - 1\}$. We call these component priority queues *subqueues*. By lemma 3.4 these subqueues use $O(t)$ space and $O(\log\log t)$ time per operation. There is one subqueue at level 1 and every leaf of every subqueue at level i has associated with it a pointer to the root of a unique subqueue at level $i + 1$, for $1 \leq i \leq k$. Let $x \in \{0,1,\ldots,m - 1\}$ and suppose $x = (d_1,d_2,\ldots,d_k)_t$. That is, d_i is the ith digit of x in base t. Each such number x describes a unique path in the structure P passing through the d_i-th leaf of the $(\sum_{j=1}^{i-1} d_j t^{i-j-1})$-th subqueue at level i, $1 \leq i \leq k$.

Suppose that at a given point in time the global priority queue contains the set of elements $S \subseteq \{0,1,\ldots,m - 1\}$. We say that a leaf of a subqueue at level i is *active* if for some $x \in S$ the path through P associated with x passes through that leaf. A subqueue at level i is *active* if it has two or more active leaves and is *trivial* if it has exactly one active leaf. The active leaves of the level k subqueues uniquely represent the elements of S.

It is easy to see that in representing the set S only the active subqueues of P are necessary. Furthermore, the trivial subqueues can be represented by a single edge linking the root with its unique active leaf. Let P' denote this modified structure. It should be clear that P' permits the execution of MEMBER, MIN, SUCCESSOR and PREDECESSOR operations in time $O(k\log\log t)$ per operation (that is, by a sequence of $O(k)$ operations in the subqueues). It remains to show how P' can be efficiently modified following an INSERT or DELETE operation. Consider the operation INSERT(x) where $x = (d_1 d_2,\ldots,d_k)_t$ (DELETE(x) is treated similarly). First d_1 is inserted into the first level subqueue. The pointer associated with the newly inserted leaf in the i-th level subqueue specifies the (i+1)-st level subqueue into which d_{i+1} is inserted. If at any level a leaf is inserted which was previously inactive then a new (trivial) subqueue is allocated at all subsequent levels. On the insertion of the second active leaf in any subqueue, that (trivial) subqueue must be replaced by an active subqueue with t leaves. As before each global operation can be accomplished with $O(k)$ subqueue operations, that is in $O(k\log\log t)$ steps. The whole point of maintaining the abbreviated structure P' rather than the conceptually simpler structure P is to save space. The structure P has a total of $O(t^{k-1}) = O(m/t)$ component priority queues each of size $O(t)$. On the other hand, since each non-trivial component priority queue in P' has at least two leaves there are at most $O(|S|)$ such components in total. The number of trivial component priority queues is certainly $O(k|S|)$, and hence the total space requirement for the structure P' is $O((k + t)|S|)$. Of course if space is not allocated dynamically then an upper bound of $O((k + t)n)$ space suffices for the maintenance of a set whose size never exceeds n.

It follows from lemma 3.4 that the subqueues can be made available with only $O(1)$ preprocessing.

4. SOME APPLICATIONS

The existence of fast algorithms for integer sorting and maintenance of integer priority queues gives rise to a number of improved algorithms for geometric problems on integer grids. We illustrate only a few here.

4.1 Convex Hulls and Related Problems

Perhaps the most straightforward application of integer sorting is to the efficient computation of convex hulls for points drawn from an integer grid in the plane. It is well known (2,8) that convex hulls can be constructed in linear time once the points have been sorted in some fashion. The following is an immediate consequence of lemma 3.1:

Theorem 4.1 The convex hull of any set of n points drawn from an m-grid in the plane can be constructed in $O(n\log\log_n m + n)$ time.

In this restricted domain the above time bound will also apply to related geometric problems whose efficient solution depends upon finding the convex hull of the set of points. For example, given the convex hull of a set of n points one can find the diameter in linear time (14).

In the real plane finding a simple closed polygonal path through n points must take $\Omega(n\log n)$ time (14).

Theorem 4.2 A simple closed polygonal path through n points drawn from an integer grid in the plane can be found in $O(n\log\log_n m + n)$ time.

Proof. Begin by dividing the set into two sets TOP and BOTTOM depending upon whether they are above or below the line joining the leftmost extreme point and the rightmost extreme point. The simple closed polygonal path through the point set is formed by joining the points in TOP in lexicographic order followed by joining the points in BOTTOM in reverse lexicographic order. Clearly the sorting step is the most time consuming; yet it falls within the time bound by lemma 3.2.

In the real plane $\Omega(n\log n)$ is a lower bound on the time required to find any triangulation of n points (15). In an integer grid domain this lower bound does not apply. The following triangulation algorithm takes advantage of grid sorting capabilities. See Figure 2.

```
1. Sort and number the points in lexicographic order.
2. Add segments to the triangulation by joining the points in sorted
   order.
3. Initialize a stack by pushing $x_1$ and then $x_2$ onto it.  Let $i = 3$.
   While  $i < n$  Do
         While (stack has two elements and the angle above the
                partially completed triangulation
                 ($x_i$, TOP(stack), SECOND(stack)) $<$ 180 degrees)
         Do
              a) Include the segment from $x_i$ to SECOND(stack) in the
                 triangulation.
              b) Pop the stack.
         End
```

```
       Push $x_i$ onto the stack.
       $i := i + 1$
       End of outer while loop
    4. Repeat step 3 considering angles below rather than above the
       partially completed triangulation.
```

Theorem 4.3 The above algorithm yields a triangulation of n points drawn from an m-grid in the plane in $O(n\log\log_n m + n)$ time.

Proof. Step 1 of the algorithm can be performed within the time bound by lemma 3.2. Step 2 is clearly linear. In steps 3 and 4 the outer while loop will take O(n) time. The inner while loop eliminates a point each time it is entered. It therefore can be entered at most n times in each of steps 3 and 4.

4.2 Subdivision Search

A grid-based rectilinear subdivision has all of its edges parallel to and integrally displaced from the standard coordinate axes. All corner points of such a subdivision lie on some m-grid in the plane. In the remainder of this paper we assume that m is polynomially related to n. This assumption enables us to use theorem 3.5 to achieve good time and space bounds simultaneously. Using a variant of the "slab" technique introduced by Dobkin and Lipton (5), it is possible to locate a point in such a subdivision in O(loglogn) time per query. The drawback of the following algorithm is that even if the storage improvements of theorem 3.5 are exploited both storage and preprocessing time are $O(n^2)$.

Begin by extending all the line segments in the subdivision to lines. This will divide the plane into $O(n^2)$ rectangular regions determined by at most n vertical and horizontal lines. Number the vertical lines from left to right and the horizontal lines from bottom to top. Use an n x n matrix A to indicate which original area in the subdivision (if any) a given newly created rectangular region lies. The coordinates of a rectangular region are determined by the numbers of the lines bounding the region on the right and above. Initialize the matrix by systematically touring through each of the original areas and determining which rectangular regions are covered by it. See Figure 3.

Next set up a priority queue to hold the O(n) x coordinates of the vertical lines. Label these coordinates with the numbers of the vertical lines that pass through them. Use the SUCCESSOR and PREDECESSOR operations in this priority queue to locate a test point with respect to the vertical lines. Use a second priority queue to locate a point with respect to the horizontal lines. It is sufficient to know the coordinates (i,j) of the rectangular region in which the test point lies. Element a_{ij} of the matrix set up in the preprocessing will contain the information as to which original area of the subdivision the test point lies.

Theorem 4.4 It is possible to locate a point within a grid-based rectilinear subdivision in O(loglogn) time and $O(n^2)$ space after $O(n^2)$ preprocessing time.

Proof. By Theorem 3.5 (with t = n), using the priority queues to find the coordinates of the rectangular region that contains the test point can be accomplished in O(loglogn) time and $O(n^2)$ space. Since there can be $O(n^2)$ of these regions initializing the matrix requires $O(n^2)$ time and space. Inserting the coordinates of the subdivision in the priority queues requires $O(n^2)$ space and O(nloglogn) time by Theorem 3.5. It is not clear whether efficient search in such restricted subdivisions based on other more space and preprocessing-time efficient schemes (10,13) is possible.

A more general type of subdivision would allow edges parallel and integrally displaced from any fixed number of axes in the plane. The algorithm for searching in a grid-based rectilinear subdivision can be generalized to work in this restricted angle subdivision without increasing the preprocessing time.

4.3 Polygonal Intersection Problems

A grid-based rectilinear polygon has all of its edges parallel to and integrally displaced from the standard coordinate axes. Based on general purpose sweeping algorithms (3,4,16,17,18) it is possible to use integer priority queues to detect intersections among grid-based rectilinear polygons.

Begin the detection algorithm by marking each vertical line segment as either left or right. Also mark each vertical line segment with the label of the polygon that it is a part of. Next sort the vertical line segments by x coordinate.

We begin a left to right sweep of the vertical line segments. If we encounter a left side we insert the y coordinates of the endpoints into an integer priority queue. We mark these points either TOP or BOTTOM and we also mark them with the label of the polygon they lie in.

Each time we make an insertion in the priority queue we perform some tests to detect intersections. Clearly if during the sweep a top corner of one polygon is between the top and bottom of another polygon the two polygons intersect. If there are a number of intersections among a set of polygons the top two polygons that intersect will have adjacent tops and the bottom two polygons that intersect will have adjacent bottoms. We can test for intersections by checking for consecutive tops or bottoms in the priority queue using the SUCCESSOR and PREDECESSOR operations. These tests will detect both edge intersections and enclosures among rectilinear polygons.

If we encounter a right side whose top endpoint is even with the top of a left side from the same polygon we delete the top of the left side from the priority queue. Otherwise insert the y coordinate of the top endpoint of the right side into the priority queue and label it BOTTOM. Its successor in the priority queue will be the top of a left side of the same polygon. Perform analogous operations with the bottom endpoint of the right side. See Figure 4.

Theorem 4.5 The above algorithm can detect whether any two of a set of grid-based rectilinear polygons intersect in O(nloglogn) time and $O(n^2)$ space.

Proof. The sort can be performed within the time and space bounds by a generalization of lemma 3.1 (11). Each vertical segment encountered necesitates priority queue operations requiring at most O(loglogn) time and $O(n^2)$ space by Theorem 3.5. Since each segment is considered at most once the proof is complete.

A more general type of polygon could be made up of sides lying parallel and integrally displaced from any fixed number of axes in the plane. The rectilinear intersection algorithm can be generalized by emplying a priority queue for each orientation to detect enclosures among a set of these more general polygons. Edge intersections among grid-based restricted angle polygons can be detected within the same time and space bounds by using an algorithm that performs a "sweep" in each of the axis orientations.

5. CONCLUSIONS

We could use some of the integer grid algorithms presented above to find fast expected-case or approximation algorithms for problems in the real plane. Consider a set of points selected from the real plane using a distribution with the property that not more than a constant number of the real points would round to the same integer. A fast expected-case algorithm for a geometric problem could round these points to integers, solve the problem in the integer grid domain and then map the solution back to the real plane. This procedure would lead to an approximation algorithm on sets of real points selected using more general distributions.

REFERENCES

1. Aho, A.V., Hopcroft, J.E. and Ullman, J.D., The Design and Analysis of Computer Algorithms, Addison-Wesley (1974).

2. Andrew, A.M., "Another Efficient Algorithm for Convex Hulls in Two Dimensions", Information Processing Letters 9,5 (1979), pp. 216-219.

3. Bentley, J.L. and Wood, D., "An Optimal Worst-Case Algorithm for Reporting Intersections of Rectangles", Technical Report, McMaster University 79-CS-13.

4. Bentley, J.L. and Ottman, Th., "Algorithm for reporting and counting geometric intersections", IEEE Trans. on Computers 26,8 (Sept. 1979)

5. Dobkin, D. and Lipton, R.J., "Multidimensional Searching Problems", SIAM J. Comput. 5,2 (June 1976), pp. 181-186.

6. van Emde Boas, P., Kaas, R. and Zijlstra, E., "Design and implementation of an efficient priority queue", Math. Systems Theory 10,2 (1977), pp. 99-127.

7. van Emde Boas, P., "Preserving Order in a Forest in Less Than Logarithmic Time and Linear Space", Information Processing Letters 6,3 (June 1977), pp. 80-82.

8. Graham, R.L., "An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set", Information Processing Letters 1, pp. 132-133.

9. Johnson, D.B., "A simple priority queue in which initialization and queue operations take O(loglogN) time", Proc. Conf. on Information Sciences and Systems, John Hopkins Univ. (1978), pp. 66-71.

10. Kirkpatrick, D.G., "Optimal Search in Planar Subdivisions". Detailed Abstract 1979.

11. Kirkpatrick, D.G., "An upper bound for sorting integers in restricted ranges", Proc. 18th Annual Allerton Conf. on Communication, Control and Computing, (Oct. 1980), pp. 243-248.

12. Knuth, D., The Art of Computer Programming, Vol. 3 (1973) Addison-Wesley.

13. Lipton, R.J. and Tarjan, R.E., "Applications of a Planar Separator Theorem", Proc. 18th Annual Symp. on Foundations of Computer Science (Oct. 1977), pp. 162-170.

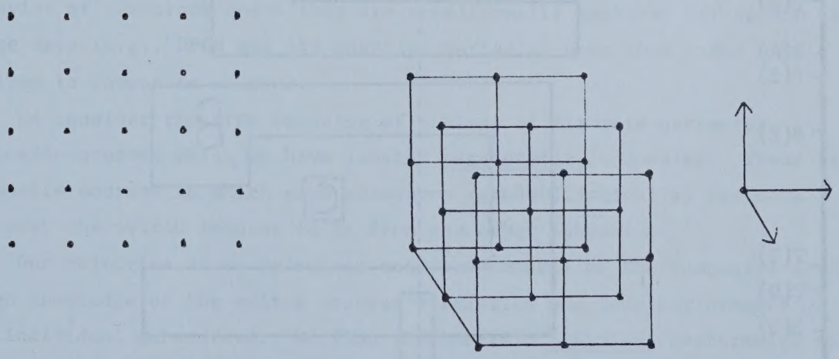
14. Shamos, M.I., "Geometric Complexity", Proc. of the Seventh Annual ACM Symp. on Theory of Computing, (May 1975), pp. 224-233.

15. Shamos, M.I. and Hoey, D., "Closest Point Problems", Proc. of the 16th Annual IEEE Symp. on Foundations of Computer Science, (Oct. 1975), pp. 151-162.

16. Shamos, M.I. and Hoey, D., "Geometric Intersection Problems", Proc. 17th Annual IEEE Symp. on Foundations of Computer Science, (1976) pp. 208-215.

17. Shamos, M.I. and Bentley, J.L., "Optimal Algorithms for Structuring Geographic Data", Proc. of an Advanced Study Symp. on Topological Data Structures for Geographic Information Systems, Harvard University, 1977.

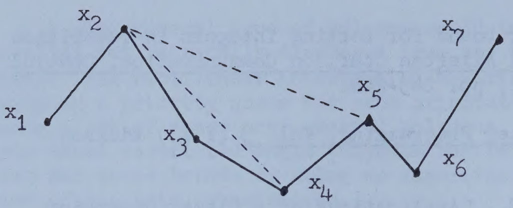
18. Vaishnavi, V., "Optimal worst-case algorithms for rectangle intersection and batched range searching problems", Technical Report, McMaster University 79-CS-12.



(a) A 5-grid in R^2.

(b) A generalized 3-grid (and unit vectors) in R^2.

Figure 1


x_1
x_2
x_3
x_4
x_5
x_6
x_7

- stack contains $x_6x_5x_2x_1$ where x_6 is at the top.
- the algorithm is about to test angle (x_7,x_6,x_5) and then include segment x_5--x_7 in the triangulation.

Figure 2 Partially Completed Triangulation

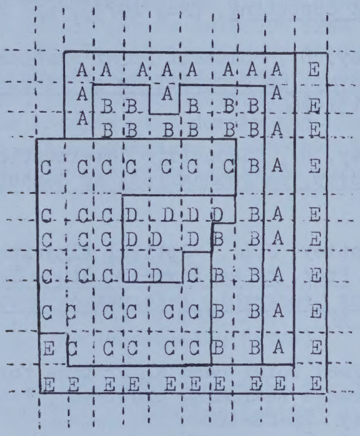


Figure 3 Locating a Point in a Rectilinear Subdivision

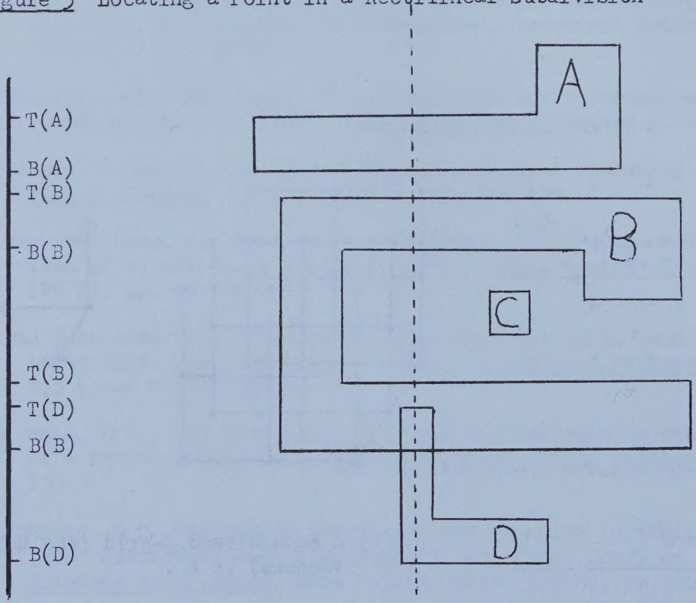

T(A)
B(A)
T(B)
B(B)
T(B)
T(D)
B(B)
B(D)
A
B
C
D

Figure 4 Detecting Intersections among Rectilinear Polygons

DPCM ENCODING OF REGENERATIVE COMPOSITE SOURCES

MICHAEL J. CARTER and DAVID L. NEUHOFF
Computer, Information and Control Engineering Program
University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

The concept of a regenerative composite source is introduced. For a selected class of such sources, it is shown that the average distortion of a DPCM code may be computed using knowledge of the switch statistics and certain transient behavior characteristics of the code. A simple approximation for the average distortion is found to compare favorably with this computational method in selected cases.

INTRODUCTION

A composite source [1] consists of an array of subsources and a switch random process. The composite source output is simply the output of the subsource currently selected by the switch. Composite source models have attracted recent attention in connection with the design of compression schemes for speech [2,3,4] and images [5,6].

A few theorems have been established concerning the performance of block source codes when applied to a particular class of composite sources [7,8,9]. However, these results are primarily asymptotic in nature (i.e., valid in the limit of long blocklengths, or as switch transitions become more infrequent). Furthermore, relatively little is known about the behavior of non-block codes that are traditionally employed for speech and image data (e.g., DPCM and its adaptive variants) when such codes are applied to composite sources.

We consider the DPCM encoding of a class of discrete-parameter composite sources which we have labeled regenerative composite. These are composite sources in which each subsource (probabilistically) restarts whenever the switch returns to it from any other subsource.

Our objective is to calculate code performance on the composite source, given knowledge of the switch process statistics and code performance on the individual subsources. We find that certain transient performance characteristics of the code can be used to compute its average distortion on a subclass of regenerative composite sources with Markov switch processes. A numerical method has been developed for evaluating average distortion. A simple approximation to the average distortion is found to compare favorably with the more complicated numerical method in several cases of interest.

REGENERATIVE COMPOSITE SOURCES

A regenerative composite source with N subsources is characterized by a switch process $\{S_j\}_{j=-\infty}^{\infty}$ with alphabet $\{1,\ldots,N\}$, and a family of stationary processes $\{Z_j^{(i)}\}_{j=-\infty}^{\infty}$, $i = 1,\ldots,N$, called modes. One mode process is associated with each subsource. The regenerative composite source works as follows. If the switch is in position i from time k to k+m , then the composite source output $X_k,\ldots,X_{k+m}$ has the same joint distribution as $Z_1^{(i)},\ldots,Z_{m+1}^{(i)}$, i.e.

$$F_{X_k,\ldots,X_{k+m}}(x_k\ldots,x_{k+m} \mid S_k = i,\ldots,S_{k+m} = i) = F_{Z_1^{(i)},\ldots,Z_{m+1}^{(i)}}(x_k,\ldots,x_{k+m}) \tag{1}$$

where $F(\cdot)$ denotes a probability distribution function. Furthermore, $X_k,\ldots,X_{k+m}$ are conditionally independent of $(\ldots,X_{j-1},X_j)$ and $(\ldots,S_{j-1},S_j)$, given that $S_j \neq i$ for some $j < k$.

The term "regenerative", as employed in this paper, does not necessarily correspond to other uses of the word in stochastic process theory. We use it simply to mean that each subsource is restarted independently of its past output whenever the switch is reconnected to it. The composite sources treated in [7,8,9] are not regenerative.

DISTORTION ANALYSIS

The simplest non-block code of interest is DPCM with a 1st order predictor in the feedback loop. Let $\{\hat{X}_j\}_{j=-\infty}^{\infty}$ denote the reproduction process (i.e., the output of the tandem connection of encoder and decoder). Then

$$\hat{X}_j = f(\hat{X}_{j-1}) + q\big(X_j - f(\hat{X}_{j-1})\big) \tag{2}$$

where $f(\cdot)$ is the predictor function and $q(\cdot)$ is the quantizer function. We note that the reproduction at time j can be expressed as $\hat{X}_j = g(X_j;\hat{X}_{j-1})$, where $g(\cdot)$ is a well defined function. The second functional argument of g is referred to as the state of the code ($\hat{X}_{j-1}$ is the state of the code at time j). For a DPCM code initialized at j=1 with state $\hat{X}_0 = x$, the code output $\hat{X}_k$ is a function of $X_1,\ldots,X_k$ and x, which we denote by $g_k(X_k,\ldots,X_1\ ;\ x)$.

The performance of a specific code is judged by its average distortion with respect to an arbitrary single-letter distortion measure d(x,y).

The rate of the code is not of immediate concern. We seek a relationship between code performance on the composite source and code performance on the underlying mode processes.

Our approach to the problem is motivated by the following observation: if one knew the state of the code and the mode type at the beginning of each mode run, and if one knew the distribution on reproduction symbols at the end of a run of length k for each k, and if one knew the switch process statistics, one could (in principle) compute the average distortion of the code on the composite source.

The average distortion, $Ed(X_j, \hat{X}_j)$ can be evaluated by first computing the average distortion conditioned on the events that: (a) the state of the code is known k time units previously, and (b) the event a switch transition to a specific subsource occurred k time units previously, and the switch has since remained connected to that subsource; one then averages over all such events. The fact that the source is regenerative composite allows us to establish the following equivalence:

$$E[d(X_j,\hat{X}_j) \mid \hat{X}_{j-k} = x, C^i_{j,k}] = E[d(Z^{(i)}_k, \hat{Z}^{(i)}_k) \mid \hat{Z}^{(i)}_0 = x] \triangleq \varepsilon^i_k(x) \tag{3}$$

where $C^i_{j,k}$ denotes the event $\{S_j = i, \ldots, S_{j-k+1} = i, S_{j-k} \neq i\}$ and $\hat{Z}^{(i)}_k = g_k(Z^{(i)}_k, \ldots, Z^{(i)}_1; x)$. The quantity $\varepsilon^i_k(x)$ is called the transient average distortion, and is the distortion incurred at time k ($k \geq 1$) when the code is applied to the $i^{\underline{th}}$ mode process, given that it is started at time 0 in initial state x.

We may now average (3) to obtain

$$Ed(X_j,\hat{X}_j) = \sum_{k=1}^{\infty} \sum_{i=1}^{N} \int \varepsilon^i_k(x)\, dF_{\hat{X}_{j-k}}(x \mid C^i_{j,k}) \cdot \Pr(C^i_{j,k}) \tag{4}$$

We note that the conditional distribution in (4) is a function only of i and not of k. Some additional notation will allow us to simplify the form of this conditional distribution. Let $\left\{T_n\right\}_{n=-\infty}^{\infty}$ be the sequence of switch transition times (i.e., T_j = least $k > T_{j-1}$ such that $S_k \neq S_{k-1}$, and without loss of generality, adjust the time origin so that $T_1 = 1$). Let $M_n = S_{T_n}$, and $W_n = \hat{X}_{T_n - 1}$. Then

$$F_{\hat{X}_{j-k}}(x \mid C^{i}_{j,k}) = F_{W_n}(x \mid M_n = i), \quad \text{for } k \geq 1 \tag{5}$$

At this point, we assume the switch process is a homogeneous, irreducible, Markov chain. One may then readily show that the pair process $\{(W_n, M_n)\}$ is homogeneous Markov. One may obtain the desired conditional distribution (5) once the stationary distribution of the pair process is known. One may then proceed to compute the average distortion using (4).

To find the stationary distribution of the pair process F_{W_n,M_n}, one may solve the following integral equation:

$$F_{W_n,M_n}(w,m) = \int F_{W_n,M_n}(w,m \mid W_{n-1} = v, M_{n-1} = \ell)\, dF_{W_n,M_n}(v\,,\,\ell) \tag{6}$$

We now wish to point out that the conditional distribution appearing as the integrand in (6) can be expressed in terms of the switch process statistics, and a collection of transient state distribution functions which characterize the operation of the code on the mode processes. To do so, we first define an auxiliary random variable $R_n = T_{n+1} - T_n$. Using the Markov property of the switch process repeatedly, one can expand the conditional distribution as shown below:

$$F_{W_n,M_n}(w,m \mid W_{n-1}=v,\, M_{n-1}=\ell) = F_{M_n}(m \mid M_{n-1}=\ell)$$

$$\cdot \sum_{k=1}^{\infty} F_{W_n}(w \mid W_{n-1}=v,\, M_{n-1}=\ell,\, R_{n-1}=k) \cdot \Pr(R_{n-1}=k \mid M_{n-1}=\ell) \tag{7}$$

We now claim

$$F_{W_n}(w \mid W_{n-1}=v\,,\, M_{n-1}=\ell,\, R_{n-1}=k) = F_{\hat{Z}_k^{(\ell)}}(w \mid \hat{Z}_0^{(\ell)}=v\,) \tag{8}$$

where the right hand side is called the transient state distribution, and is the probability distribution function of the reproduction process at time k when the code is applied to the ℓ^{th} mode process, given that the code is started in state $\hat{Z}_0^{(\ell)} = v$. The collection of such distributions for all $(v\,,k)$ describes the behavior of the code when applied to a given mode process.

We conclude this section by observing that we have shown code performance on the composite source to be related to code performance on the underlying mode processes using the concepts of transient average distortion, and transient state distributions.

COMPUTATIONAL PROCEDURE FOR AVERAGE DISTORTION

The procedure outlined in the previous section requires that we know the transient average distortion and transient state distribution characteristics of the code for each mode process. While one could (in principle) find an analytical formula for each of the transient characteristics, it would probably be impossible to solve (6) analytically for the pair process distribution. The evaluation of the integral in (4) by other than numerical methods would present similar difficulties. In this section, we describe a computational procedure for obtaining the average distortion of the code when applied to the composite source.

We first approximate $\varepsilon_k^i(x)$ and $F_{\hat{Z}_k^{(i)}}(\cdot \mid \hat{Z}_0^{(i)} = x)$ as follows:

(a) Encode an ensemble of pseudorandomly generated sequences of length L, typical of the $i^{\underline{th}}$ mode process, starting with initial code state x selected from a finite set of values $\{x_1, \ldots, x_{N_Q}\}$. These values are the output levels of a uniform quantizer $Q(\cdot)$ with a large number, N_Q, of levels.

(b) For each x_j and each mode i, compute the sample average distortion over the ensemble at each time k. This approximates $\varepsilon_k^i(x_j)$.

(c) Using $Q(\cdot)$, quantize the encoded sequences. For each x_j and each mode i, construct a histogram for $Q(\hat{Z}_k^{(i)})$ at each time k. For any $x_h \varepsilon \{x_1, \ldots, x_{N_Q}\}$, $P^{(i)}(h,j,k)$ denotes the probability that $Q(\hat{Z}_k^{(i)})$ equals x_h, given that $\hat{Z}_0^{(i)} = x_j$. The distribution function corresponding to this histogram approximates $F_{\hat{Z}_k^{(i)}}(\cdot \mid \hat{Z}_0^{(i)} = x_j)$.

(d) For k>L, we assume that $\varepsilon_k^i(x)$ and $F_{\hat{Z}_k^{(i)}}(\cdot \mid \hat{Z}_0^{(i)} = x_j)$ are essentially the same as at time k = L. We have found experimentally that this is true when L is large enough (L = 20 typically).

Some theoretical justification for this assumption is provided in [10].

Consider the pair process $\left\{(Q(W_n),M_n)\right\}$ which has $N_Q \times N$ states. Let π be the row vector of stationary probabilities corresponding to these states. Let P_T be the corresponding matrix of transition probabilities for the state process. Then π must satisfy

$$\pi = \pi P_T \tag{9}$$

We compute entries for P_T using the array P as shown below:

$$\begin{aligned} P_T[(h,m) \mid (j,\ell)] &= \Pr(M_n = m \mid M_{n-1} = \ell) \\ &\cdot \left\{ \sum_{k=1}^{L-1} P^{(m)}(h,j,k) \cdot \Pr(R_{n-1} = k \mid M_{n-1} = \ell) \right. \\ &\left. + P^{(m)}(h,j,L) \cdot \Pr(R_{n-1} \geq L \mid M_{n-1} = \ell) \right\} \end{aligned} \tag{10}$$

We then solve for π using (9). The distribution function corresponding to π approximates F_{W_n,M_n}.

The remainder of the computational procedure is straightforward, and simply involves the use of the approximations to $\varepsilon_k^i(\cdot)$ and $F_{W_n|M_n}(\cdot|\cdot)$ in the evaluation of (4). We note that assumption (d) allows the infinite summation in (4) to be truncated. A computer program which implements the procedure just described has been written. Average distortions computed using this method are in very good agreement with values computed from encodings of pseudorandomly produced regenerative composite sequences.

SIMPLE APPROXIMATION

Our numerical method can be computationally expensive, and therefore one is motivated to find an approximation to the average distortion which is more readily evaluated. A very simple estimator for the average distortion is suggested by the following intuitive observation. If the average mode runlengths are long (i.e., the switch transitions are infrequent), then the code operates on each subsource for a long time. We expect that, after an initial transient period, the performance of the code is quite close to its asymptotic distortion for the corresponding mode process. When encoding the composite source, the average distortion of the code should then be well approximated by

$$\sum_{i=1}^{N} \Pr(S_j = i) \cdot Ed(Z_j^{(i)}, \hat{Z}_j^{(i)}) \tag{11}$$

i.e., weight the average distortion on the i^{th} mode process by the fraction of time the switch spends on the i^{th} subsource, and sum over all modes.

We have briefly studied the accuracy of this approximation as compared with the results of the more exact numerical method. The composite source selected for study has two different Gauss-Markov mode processes, and a binary symmetric Markov switch process. The mode processes each have unit variance, and correlation coefficients ρ_1, ρ_2 respectively. The switch has transition probability p, and average mode runlengths are therefore both 1/p. The DPCM code is specified by

$$\hat{X}_j = \tilde{\rho}\hat{X}_{j-1} + q(X_j - \tilde{\rho}\hat{X}_{j-1}) \tag{12}$$

where $q(\cdot)$ is a two level quantizer with output levels ± qmax. The distortion measure is squared error.

Figure 1 shows the percentage difference between the average distortion estimate (11) (which we denote by D') and the numerical method result (called D) as a function of average mode runlength. The results agree closely when the average mode runlengths are moderate (20 time units) to long (> 100 time units). Studies of DPCM codes with other predictor coefficients and quantizers confirm the accuracy of the simple approximation when the switch changes position slowly.

When code (a) is employed, the approximation result differs significantly from the more exact result. A probable cause for the disagreement noted at short average mode runlengths may be inferred from examination of the transient average distortion sequences of the code for each mode. Figure 2 shows typical transient average distortion sequences of code (a) when applied to each mode. When the average mode runlengths are short, the code operates in the transient region of the curves (cf. Fig. 2) much of the time. The transient average distortion in this region is significantly worse than the asymptotic distortion. This produces the disagreement in results between the two methods at short (< 20 time units) average mode runlengths.

A surprising result is that in certain cases, the simple approximation agrees closely with the numerical method at very short average mode runlengths (cf. Fig. 1, code (b)). In such instances, we note that the transient average distortion sequences deviate little from their asymptotic values. Figure 3 shows typical transient average distortion sequences for code (b) when applied to the same modes already considered. We conjecture that when the code is mismatched to both mode processes (i.e., not optimized for least asymptotic distortion), there is an absence of significant transient behavior. The code very nearly achieves its asymptotic distortion all the time, and therefore the simple approximation performs well at all average mode runlengths.

REFERENCES

[1] T. Berger, Rate Distortion Theory, Englewood Cliffs, NJ: Prentice-Hall, 1971 (see Chapter 6).

[2] R.M. Gray, A. Buzo, Y. Matsuyama, A.H. Gray, Jr., and J.D. Markel, "Source Coding and Speech Compression," Proc. of the Internatl. Telemetering Conference, vol. 14, pp. 871-878, 1978.

[3] V. Ramamoorthy, "Speech Coding Based on a Composite Gaussian Source Model," Proc. of 1979 Int. Conf. on ASSP, Washington, D.C.: IEEE Press, pp. 534-537, 1979.

[4] Y. Matsuyama and R.M. Gray, "Universal Tree Encoding for Speech," IEEE Trans. Inform. Theory, vol. IT-27, pp. 31-40, Jan. 1981.

[5] I.J. Dukhovich, "Composite Model of a 3-D Image," Proc. of SPIE, vol. 249, Advances in Image Transmission II, 1980.

[6] I.J. Dukhovich, "A DPCM System Based on a Composite Image Model," Proc. 1980 National Telecomm. Conf., Houston, TX: IEEE Press, pp. 36.2.1 - 36.2.6 , 1980.

[7] R.J. Fontana, "Limit Theorems for Slowly Varying Composite Sources," IEEE Trans. Inform. Theory, vol. IT-26, pp. 702-709, Nov. 1980.

[8] R.J. Fontana, "Universal Codes for a Class of Composite Sources," IEEE Trans. Inform. Theory, vol. IT-26, pp. 480-482, July 1980.

[9] S.A. Garde, "Communication of Composite Sources," Ph.D. Dissertation, University of California, Berkeley, Berkeley, CA, 1980.

[10] A. Gersho, "Stochastic Stability of Delta Modulation," Bell System Tech. J., vol. 51, No. 4, pp. 821-841, Apr. 1972.

The work reported herein was supported by the National Science Foundation.

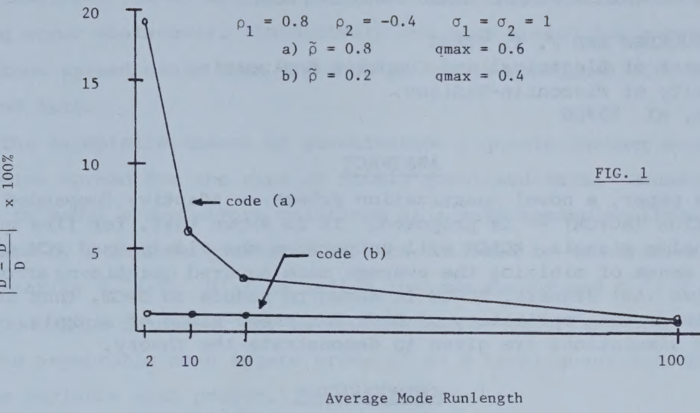


FIG. 1

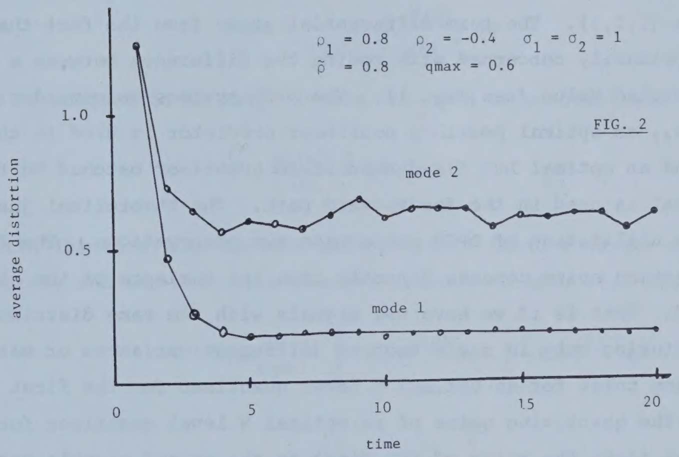


FIG. 2

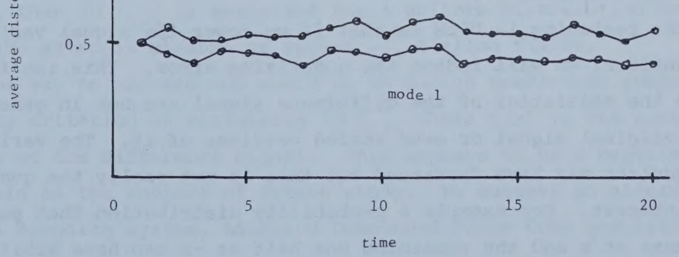


FIG. 3

ADAPTIVE COMPANDED PULSE-CODE-MODULATION

J. A. BUCKLEW AND F. T. TZENG
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Madison, WI 53706

ABSTRACT

In this paper, a novel quantization scheme -- Adaptive Companded Pulse Code Modulation (ACPCM) -- is proposed. It is shown that, for fine quantization of analog signals, ACPCM will outperform the widely-used PCM and DPCM in the sense of minizing the average mean squared quantizong error. For autoregressive (AR) signals, ACPCM is shown to reduce to DPCM, thus demonstrating a large rate optimality of DPCM for these kinds of signals. Some examples and simulations are given to demonstrate the theory.

I. INTRODUCTION

Differential Pulse Code Modulation (DPCM) has been investigated by several authors [1,2,3]. The term differential stems from the fact that the encoder is primarily concerned with coding the difference between a sample and its predicted value (see Fig. 1). The DPCM systems we consider are optimized, i.e., an optimal possibly nonlinear predictor is used in the feedback loop and an optimal but fixed nonuniform quantizer matched to the difference signal is used in the feedforward path. The theoretical justification for the utilization of DPCM rests upon two observations. The first is that quantization noise depends directly upon the variance of the signal to be quantized. That is if we have two signals with the same distribution function differing only in scale factors (different variances or means) then the quantizing noise for an optimal N level quantizer for the first signal is equal to the quantizing noise of an optimal N level quantizer for the second signal times the ratio of the first to the second signals variance. The second theoretical reason for using DPCM is that in many typical source encoding applications the sample to sample correlation is fairly high. A predictor can then be implemented, utilizing information from previous quantized samples, with difference signal variance significantly smaller than the original sample variance. The DPCM system then quantizes this difference signal and transmits it.

Hence the reasoning in DPCM is that if we lower the signal variance into the quantizer, we will reduce the quantizing error. This isn't quite true because the statistics of the difference signal are not in general the same as the original signal or even scaled versions of it. The variance into the quantizer may have decreased but this is not really the quantity of primary interest. For example a probability distribution that puts one half of its mass at n and the remaining one half at -n can have arbitrarily large variance but an optimal two level quantizer will code it with no quantizing error whatsoever. Intuitively one sees a need for a measure of distribution spread other than variance if one is concerned with the quantization of data.

The asymptotic theory of quantization suggests another measure of distribution spread for the case of finely quantized data. Bennett [4] was the first to model a nonuniform quantizer as a zero memory nonlinearity $G(\cdot)$ followed by a uniform quantizer in turn followed by the inverse of the first nonlinearity $G^{-1}(\cdot)$. $G(\cdot)$ is called the compressor and $G^{-1}(\cdot)$ the expandor. Any nonuniform quantizer may be represented in this fashion. His expression for the asymptotic mean square error of an N level quantizer operating on a random variable with probability density $f(x)$ is

$$\text{mse} \cong \frac{1}{12N^2} \int \frac{f(x)}{g(x)^2}\,dx \tag{1}$$

where N is assumed to be large (hence the term asymptotic) and

$$\frac{dG(x)}{dx} \triangleq g(x).$$

Smith [5] took this formula and minimized it to find the optimum nonlinearity and the mean squared error associated with it. His results are

$$G(x)_{\text{opt}} = \int_{-\infty}^{x} f(\alpha)^{1/3}d\alpha \Big/ \int_{-\infty}^{\infty} f(\alpha)^{1/3}d\alpha \tag{2}$$

and

$$\text{mse}_{\text{opt}} = \frac{\left(\int_{-\infty}^{\infty} f(\alpha)^{1/3}dx\right)^3}{12N^2} \triangleq \frac{\|f\|_{1/3}}{12N^2} \tag{3}$$

The above equation implies that for the purposes of finely quantizing data $\|f\|_{1/3}$ is the measure of spread we desire to use. We remark in passing that for $f(x)$ constrained to be of finite support on some interval $[a, b]$, then $\|f\|_{1/3}$ is maximized for a uniform distribution over $[a, b]$ as is the entropic measure of spread $-\int_a^b f(x)\log f(x)dx$.

One way to proceed now would be to design predictors subject to a fidelity criterion of minimizing $\|f\|_{1/3}$ where $f(x)$ is the probability density of the difference signal. This appears to be a nontrivial problem and could be the subject of future study. We suggest an alternate method, a novel encoding system, Adaptive Companded Pulse Code Modulation (ACPCM) (see Fig. 2). Instead of a predictor in a feedback loop trying to minimize some measure of spread of the difference signal, we change the shape of the compressor function dynamically with time.

In the next section we give arguments for the large rate optimality of this encoding system. As a consequence of this theory we can show that for autoregressive signals (AR) ACPCM reduces to DPCM and hence DPCM is optimal for finely quantizing AR signals.

In the last section we provide some computer simulation results comparing the performance of Pulse Code Modulation (PCM), DPCM and ACPCM.

II. DEVELOPMENT

Suppose we wish to quantize with $\log_2 N$ bits per sample the data sequence $\{y_n\}$. Denote $(y_{n-1}, y_{n-2}, \ldots, y_{n-k}) \overset{\Delta}{=} \underline{y}_{n-k}$. Let $p_{y_n}(\cdot)$ and $p_{y_n|\underline{y}_{n-k}}(\cdot|y_{n-k})$ respectively denote the probability densities of the random variable y_n and the random variable y_n conditioned on the random vector $\underline{y}_{n-k}$. A consideration of equation (1) indicates that the best compressor characteristic for $p_{y_n|\underline{y}_{n-k}}(\cdot|\underline{y}_{n-k})$ is

$$G(y_n|\underline{y}_{n-k}) = \int_{-\infty}^{y_n} p^{1/3}_{y_n|\underline{y}_{n-k}}(u|\underline{y}_{n-k})du \Big/ \int_{-\infty}^{\infty} p^{1/3}_{y_n|\underline{y}_{n-k}}(u|\underline{y}_{n-k})du \, . \tag{4}$$

This equation indicates that when we have access to information about previous samples we should have a time varying compressor (and expandor) nonlinearity matched to the variations in those previous samples. Since the compressor adapts itself to the previous data statistics we will call a quantization scheme making use of these time varying nonlinearities, Adaptive Companded Pulse Code Modulation (ACPCM).

We remark that in actuality we need to use the previous k quantized samples $\hat{\underline{y}}_{n-k}$ instead of $\underline{y}_{n-k}$ in the above equation since the receiver with its time varying expandor (at possibly a remote location) must be able to "track" anything the ACPCM encoder does. But since we are considering only finely quantized data, $p_{y_n|\underline{y}_{n-k}}(y_n|\hat{\underline{y}}_{n-k}) \approx p_{y_n|\underline{y}_{n-k}}(y_n|\underline{y}_{n-k})$ and with little loss in generality we will continue our present notation. We note also that if y_n is independent of $\underline{y}_{n-k}$ then $p_{y_n|\underline{y}_{n-k}}(\cdot|\underline{y}_{n-k}) = p_{y_n}(\cdot)$ and ACPCM is equivalent to PCM with Equations (1-3) applying.

Differential Pulse Code Modulation (DPCM) systems (see Fig. 1) also try to make use of the information in previous samples to lower quantization noise. In these systems we are quantizing a difference signal that is the difference between the sample value y_n and the predicted value $M(y_n|\underline{y}_{n-k})$ of y_n based upon $\underline{y}_{n-k}$. For the case of a minimum variance estimate $M(y_n|\underline{y}_{n-k}) = E\{y_n|\underline{y}_{n-k}\}$. For the case of PCM we can view the prediction operation as setting $M(y_n|\underline{y}_{n-k}) \overset{\Delta}{=} 0$. The remainder of this section will be concerned with comparing the performance of ACPCM with an arbitrary feedback predictor system ($M(y_n|\underline{y}_{n-k})$ is allowed to be any function of $\underline{y}_{n-k}$).

Let $D_n \overset{\Delta}{=} y_n - M(y_n|\underline{y}_{n-k})$. A consideration of Eq. (3) indicates that for finely quantized data a good approximation to the mean squared error performance of ACPCM is

$$mse_{ACPCM} = \frac{1}{12N^2}\int\left[\int p^{1/3}_{y_n|\underline{y}_{n-k}}(y_n|\underline{y}_{n-k})dy_n\right]^3 p(\underline{y}_{n-k})d_{\underline{y}_{n-k}} .$$

Let $D_n = y_n - M(y_n|\underline{y}_{n-k})$ implies

$$= \frac{1}{12N^2}\int\left[\int_{p_{y_n\,\underline{y}_{n-k}}} p^{1/3}_{y_n|\underline{y}_{n-k}}(D_n + M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})dD_n\right]^3 p(\underline{y}_{n-k})d\underline{y}_{n-k} \qquad (5)$$

Note that $M(y_n|\underline{y}_{n-k})$ is not a function of y_n. For an optimal differential pulse-code system the compressor is fixed and is matched to the statistics of the difference signal. The probability density function of the difference signal D_n is

$$\int p_{y_n|\underline{y}_{n-k}}(D_n + M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})p_{\underline{y}_{n-k}}(\underline{y}_{n-k})d\underline{y}_{n-k} \overset{\Delta}{=} p_{D_n}(D_n).$$

Hence, from Eq. (3), the mean squared error for such an otpimal feedback predictor system is

$$mse_{DPCM} = \frac{1}{12N^2}\left[\int\left(\int p_{y_n|\underline{y}_{n-k}}(D_n + M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})p_{\underline{y}_{n-k}}(\underline{y}_{n-k})d\underline{y}_{n-k}\right)^{1/3} dD_n\right]^3$$

$$= \frac{1}{12N^2}\left\{\int\left(\frac{\int p_{y_n|\underline{y}_{n-k}}(D_n+M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})p_{\underline{y}_{n-k}}(\underline{y}_{n-k})d\underline{y}_{n-k}}{(\int p_{y_n|\underline{y}_{n-k}}(D_n+M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})p_{\underline{y}_{n-k}}(\underline{y}_{n-k})d\underline{y}_{n-k})^{2/3}}\right)dD_n\right\}\left(\int p^{1/3}_{D_n}(D_n)dD_n\right)^2$$

$$= \frac{1}{12N^2}\int\left(\frac{\int p_{y_n|\underline{y}_{n-k}}(D_n+M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})p_{\underline{y}_{n-k}}(\underline{y}_{n-k})d\underline{y}_{n-k}}{(p^{1/3}_{D_n}(D_n)/\int p^{1/3}_{D_n}(D_n)dD_n)^2}\right)dD_n$$

$$= \frac{1}{12N^2}\int\left(\int\frac{p_{y_n|\underline{y}_{n-k}}(D_n+M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k}}{(p^{1/3}_{D_n}(D_n)/\int p^{1/3}_{D_n}(D_n)dD_n)^2}dD_n\right)p_{\underline{y}_{n-k}}(\underline{y}_{n-k})d\underline{y}_{n-k} .$$

Note that $p^{1/3}_{D_n}(D_n)/\int p^{1/3}_{D_n}(D_n)dD_n$ is a probability density function. In [5] it is shown that

$$\min_{\lambda(x)} \int \frac{p(x)}{\lambda(x)^2}\, dx \triangleq \left(\int p(x)^{1/3} dx\right)^3 \triangleq \|p\|_{1/3}$$

where $p(x)$ is a probability density and the minimization is over all probability densities $\lambda(x)$. Hence

$$mse_{DPCM} \geq \frac{1}{12N^2} \int \left(\int p^{1/3}_{y_n|\underline{y}_{n-k}} (D_n + M(y_n|\underline{y}_{n-k})|\underline{y}_{n-k})\, dD_n \right)^3 p(\underline{y}_{n-k})\, d_{\underline{y}_{n-k}} = mse_{ACPCM}$$

by Eq. (5).

This inequality indicates that no matter what kind of predictor is used in the feedback loop of a DPCM type system, ACPCM will do at least as well.

For stationary autoregressive type signals, $y_n = py_{n-1} + x_n$ where $\{x_n\}$ is an independent identically distributed random sequnce and $p \in [0,1]$, we have

$$p_{y_n|\underline{y}_{n-k}}(y_n|\underline{y}_{n-k}) = p_{x_n}(y_n - py_{n-1}).$$

Also for a DPCM system $D_n = y_n - E\{y_n|\underline{y}_{n-k}\} = y_n - py_{n-1} = x_n$. This implies that

$$mse_{ACPCM} = \int \frac{\left(\int p_{x_n}(y_n - py_{n-1})^{1/3} dy_n\right)^3}{12N^2}\, p_{y_n}(y_{n-1})\, dy_{n-1}$$

$$= \left(\int p_{x_n}(y_n)^{1/3} dy_n \right)^3 / 12N^2 = mse_{DPCM}$$

Hence for these types of signals DPCM is the best possible way to incorporate previous sample information.

III. EXAMPLES AND THE SIMULATION RESULTS

In this section, we present two examples. The first example is an additive Gaussian Markov model, in which the data samples are correlated. As proved in the previous section ACPCM is identical to DPCM, and is known to have better performance than PCM. The second example is a multiplicative Gaussian Markov model, in which the data samples are uncorrelated. It is shown that, for this model, DPCM is identical to PCM, and both are inferior to ACPCM.

(I) Additive Gaussian Markov Model $y_n = py_{n-1} + x_n$

Here, x_n is Gaussian distributed with $p_{x_n}(x_n) = \frac{1}{\sqrt{2\pi}} = e^{-x_n^2/2}$. Let p be a real number in $[0, 1)$. Since $y_n = py_{n-1} + x_n$, it is clear that y_n is also a Gaussian distribution, with $E(y_n) = 0$ and $E(y_n^2) = 1/1-p^2$.

(A) ACPCM (DPCM). In Section II, we had shown that for this model, ACPCM is identical to DPCM. Since $p_{y_n|y_{n-1}}(y_n|y_{n-1}) = p_{x_n}(y_n - py_{n-1})$,

$$MSE_{ACPCM} = \frac{1}{12N^2}\left\{\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\left(\frac{1}{\sqrt{2\pi}}\, e^{-x_n^2/2}\right)^{1/3} dx_n\right]^3 p(y_{n-1})dy_{n-1}\right\}$$

$$= \frac{\sqrt{3}}{2}\,(\pi/N^2)$$

Assume we use a 7 bits/sample data rate which implies $N = 123$.

$$\therefore\ MSE_{ACPCM} = 1.66 \times 10^{-4} \tag{7}$$

Note this result is independent of p.

(B) PCM. Since $p_{y_n}(y_n) = \frac{1}{\sqrt{2\pi}\,\sigma}\, e^{-y_n^2/2\sigma^2}$ where $\sigma^2 = 1/1-p^2$

$$\therefore\ MSE_{PCM} = \frac{1}{12N^2}\left[\int_{-\infty}^{\infty} p_{y_n}^{1/3}(y_n)dy_n\right]^3 \tag{8}$$

$$= 1.661 \times 10^{-4}\left(\frac{1}{1-p^2}\right)$$

(C) Comparison and the Simulation Results. Eqs. (7), (8) clearly indicates that when the correlation coefficient $p \neq 0$, ACPCM will always have better performance than PCM. Further, as p gets larger, the advantages of using ACPCM are more and more obvious.

Fig. 3 shows the theoretical and the simulation results for both systems. The simulation results are derived by using 1000 data samples generated from this model and a Gaussian random number generating routine as inputs to both systems. They are in good agreement with the theoretical values calculated above.

(II) Multiplicative Gaussian Markov Model $y_n = y_{n-1}^p x_n$

Here, x_n is also a Gaussian distribution $N(0,1)$ and p is a real number in $[0,1]$. Since $E(y_n y_{n-1}) = E(y_{n-1}^{1+p} x_n) = E(y_{n-1}^{1+p})E(x_n) = 0$ and $E(y_n) = E(y_{n-1}^p x_n) = 0$, we have $E(y_n y_{n-1}) = E(y_n)E(y_{n-1})$, i.e., in this model, y_n and y_{n-1} are uncorrelated. Further, $E(y_n|y_{n-1}) = E(y_{n-1}^p x_n|y_{n-1}) = y_{n-1}^p E(x_n) = 0$, so the difference signal for DPCM is $D_n = y_n - E(y_n|y_{n-1}) = y_n$ which is identical to PCM. In the following, we show that for this uncorrelated but still dependent case, ACPCM is better than DPCM and PCM.

(A) ACPCM

$$p_{y_n|y_{n-1}}(y_n|y_{n-1}) = p_{x_n}(y_n/y^p_{n-1}) \frac{1}{|y^p_{n-1}|}; \quad p_{x_n}(x_n) = \frac{1}{\sqrt{2\pi}}{}^{-x_n^{2}/2}$$

$$\therefore \ \text{MSE}_{\text{ACPCM}} = \frac{1}{12N^2}\left[\int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty} p^{1/3}_{y_n|y_{n-1}}(y_n\ y_{n-1})dy_n\right)^3 p(y_{n-1})dy_{n-1}\right] \qquad (9)$$

$$= 1.661 \times 10^{-4} \int y^{2p}_{n-1}p(y_{n-1})dy_{n-1}$$

$$= 1.661 \times 10^{-4}\ \overline{y^{2p}_{n-1}}$$

(B) DPCM (PCM). For this model, we were unable to derive the analytical density function of y_n. Hence, we approximate the density function of y_n by using 20,000 data samples and calculating a histogram from which we calculate the following integration

$$\text{MSE}_{\text{DPCM}} = \frac{1}{12N^2}\left(\int_{\infty}^{\infty} p^{1/3}_{y_n}(y_n)dy_n\right)^3 .$$

These 20,000 data samples also give us our value for $\overline{y^{2p}_{n-1}}$. Substitute this value into Eq. (9) and we may get the mean square error of ACPCM for each value of p.

Fig. 4 shows the theoretical mean square errors of both ACPCM and DCPM. Due to truncation error in the numerical integration, the mean squared error of DPCM is shown to be a little less than that of ACPCM when p=0. When p=0 they of course should be equal. The mean square error curve of DPCM nevertheless crosses this gap and climbs over the mean square error curve of ACPCM as p gets larger. Interestingly enough, as p increases the signal to quantizing noise ratio increases for ACPCM and decreases for DPCM.

Also shown in Fig. 4 is the simulation results for ACPCM. This data is the average of ten runs, each with 3,000 samples and a different seed for the Gaussian random number generating routine. This result is also within good agreement to the theoretical result calculated above.

IV. SUMMARY

In this paper we reconsider the philosophy of differential pulse code modulation (DPCM). We suggest that the variance measure of distributional spread for the purposes of finely quantizing data in an inappropriate one. We present an alternate measure of spread derived from asymptotic quantization theory. We present a system Adaptive Compressive Pulse Code Modulation (ACPCM) based upon Bennett's companding model to minimize this different spread measure. We derive an inequality which demonstrates that ACPCM will always outperform DPCM when the data is to be finely quantized. We show that for stationary autoregressive type signals that DPCM and ACPCM give identical performance. We present some examples and simulation results to illustrate the theory.

V. REFERENCES

[1] J. B. O'Neal and R. W. Stoh, "Differential PCM for Speech and Data Signals," IEEE Trans. Comm., Vol. COM-20, pp. 900-912, Oct. 1972.

[2] N. S. Jayant, "Digital Coding of Speech Waveforms: PCM, DPCM, and DM Quantizers," Proc. IEEE, Vol. 62, pp. 611-632, May 1974.

[3] J. B. Anderson and J. B. Bodie, "Tree Encoding of Speech," IEEE Trans. Inform. Theory, Vol. IT-21, pp. 379-387, July 1975.

[4] W. R. Bennett, "Spectra of Quantized Signals," BSTJ, Vol. 27, pp. 446-472, July 1948.

[5] B. Smith, "Instantaneous Companding of Quantized Signals," BSTJ, Vol. 36, pp. 653-709, May 1957.

[6] P. F. Panter, W. Dite, "Quantization Distortion in Pulse-Count-Modulation With Nonuniform Spacing of Levels," Proc. IRE, Vol. 39, pp. 44-48, Jan. 1951.

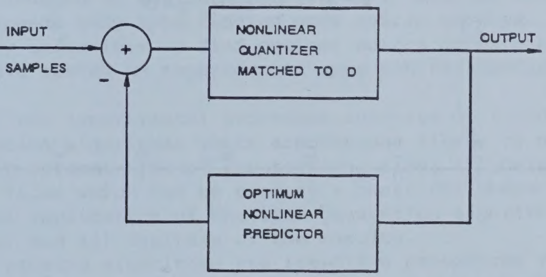

INPUT
SAMPLES
+
-
NONLINEAR QUANTIZER MATCHED TO D
OUTPUT
OPTIMUM NONLINEAR PREDICTOR

FIG. 1: AN OPTIMUM DIFFERENTIAL PULSE CODE MODULATOR

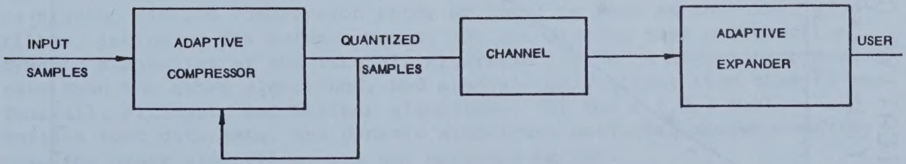

INPUT
SAMPLES
ADAPTIVE COMPRESSOR
QUANTIZED
SAMPLES
CHANNEL
ADAPTIVE EXPANDER
USER

FIG. 2: ADAPTIVE COMPRESSIVE PULSE CODE MODULATION SYSTEM BLOCK DIAGRAM

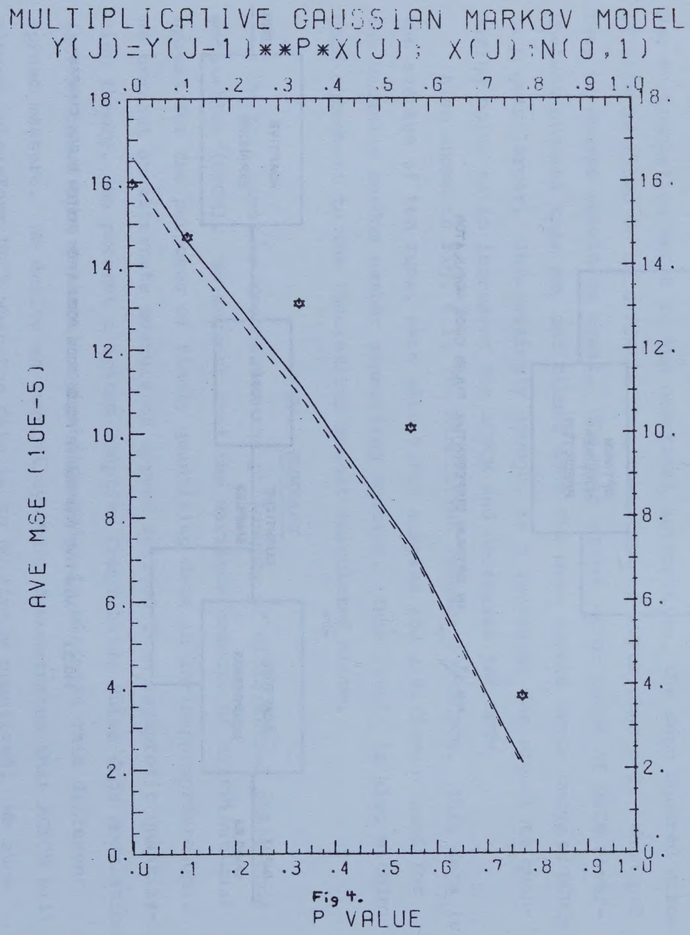

MULTIPLICATIVE GAUSSIAN MARKOV MODEL
Y(J)=Y(J-1)**P*X(J); X(J):N(0,1)
AVE MSE (10E-5)
P VALUE
.0
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
0.
2.
4.
6.
8.
10.
12.
14.
16.
18.

Fig 4.

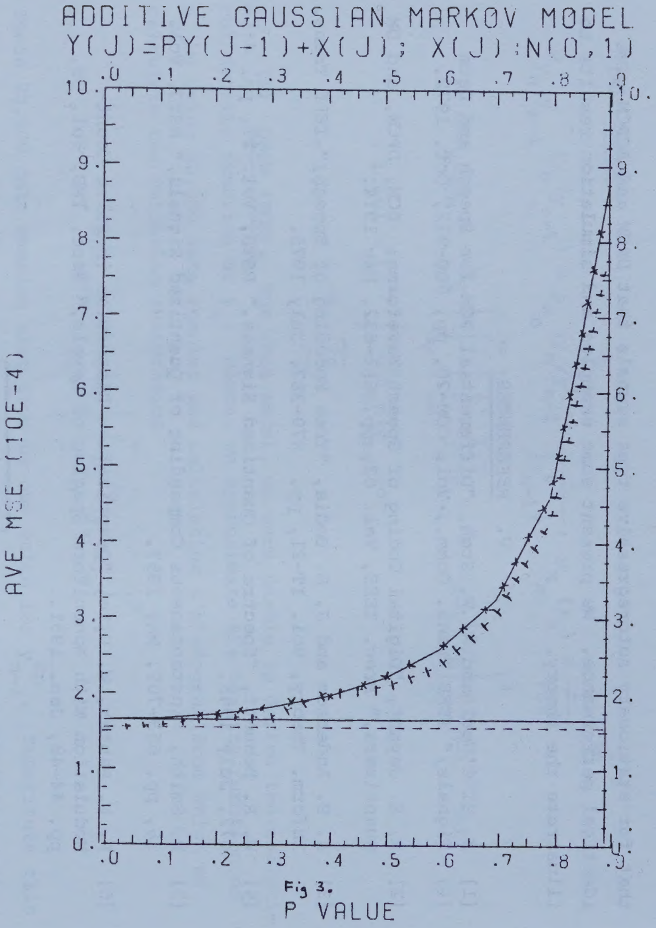

ADDITIVE GAUSSIAN MARKOV MODEL
Y(J)=PY(J-1)+X(J); X(J):N(0,1)
AVE MSE (10E-4)
P VALUE
.0
.1
.2
.3
.4
.5
.6
.7
.8
.9
0.
1.
2.
3.
4.
5.
6.
7.
8.
9.
10.

Fig 3.

AN EXPERIMENTAL APPROACH FOR DESIGNING VARIABLE-LENGTH TO VARIABLE-LENGTH CODES*

NORMAN R. KOLB
Western Electric Company
Ballwin, Missouri 63011

JAMES G. DUNHAM
Department of Electrical Engineering
Washington University
St. Louis, Missouri 63130

ABSTRACT

Efficient procedures exist for designing Huffman and Tunstall codes for a given stationary source model, but to date little is known about the design of optimal variable-length to variable-length (VL-VL) codes. The difficulty of the problem is compounded by the fact that an adequate model may not be known for sources encountered in practical data compression applications. Without the model, efficient codes and estimates of compression bounds are difficult to obtain.

A VL-VL code can be viewed as a pair of back-to-back rooted binary trees with the same number of terminal nodes, and is denoted by (S,R). The first tree, S, is called the source parser and maps a variable number of source symbols into intermediate symbols. The other tree, R, is called the channel parser and maps the intermediate symbols into a variable number of channel symbols.

A theorem established by Dunham states that S determines the potential compression while R attempts to achieve this compression. Thus, to a first order, the estimation of source compressibility and the design of good VL-VL codes depends only upon finding good source parsers. Since it is computationally infeasible to find optimal source parsers by exhaustive testing, we have tested an experimental approach for finding good source parsers.

In brief, the experimental procedure consists of (1) developing a set of tree generation algorithms which are thought likely to generate the optimal or nearly optimal tree of the required size; (2) developing a set of standard data files which can be used as a basis for comparison of the algorithms; (3) application of the tree generation algorithms to the standard data sets; and (4) analysis of the results.

The tree growing algorithms are iterative procedures which make node growing decisions from dynamically measured source statistics. The source string is parsed (perhaps only partially) using the tree in its current state, and the codeword counts are used as the source statistics. For example, one algorithm, similar to Tunstall's algorithm, extends the branch whose node count is highest. In the experimental procedure, the performance achieved with these algorithms is compared to that of the Tunstall, Fitingof, and Huffman algorithms.

The principal results of the comparison of experimental data are as follows: (1) in all trials except the ECG data sets, the dynamic algorithms as a group yielded compression rates at least as high as the other algorithms, including the Huffman codes; (2) for digital test patterns the dynamic adaptation of the Tunstall algorithm yielded a higher compression rate than the other algorithms, and significantly higher than that of the Tunstall, Fitingof, and Huffman algorithms; (3) for i.i.d., Markov, and English text data sets, the dynamic algorithms performed somewhat better than the other algorithms, but not markedly better.

The experimental results verify that the dynamic adaptation of the Tunstall algorithm is a viable method of building VL-FL trees suitable for estimating compressibility bounds and constructing good VL-VL codes.

---

*This work was supported in part by NSF Grant ENG-7910001 and Grant RR-00396 from the Division of Research Resources, NIH.

INFORMATION THEORY AND CAUSAL INFORMATION TRANSMISSION WITH FEEDBACK

TOBY BERGER
School of Electrical Engineering
Cornell University
Ithaca, NY 14853

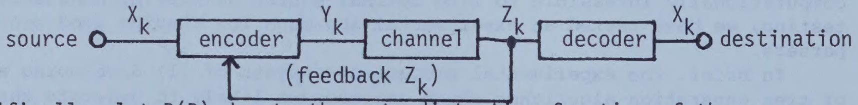
GARY MUNSON
Center for Applied Mathematics
Cornell University; and
Bell Laboratories
Holmdel, NJ 07733

ABSTRACT

For a given source, channel and fidelity criterion, Gorbunov and Pinsker [1], and Teneketzis [2] have provided information-theoretic lower bounds to the value of least average distortion attainable in source data reproduction by any causal code. Here it is shown that, for any independent, identically distributed (iid) source, neither bound improves upon that already provided by classical information theory. An alternate maximization formulation of the Gorbunov-Pinsker bound is given which more readily facilitates lower bounding their lower bound, which is usually difficult to calculate precisely. Another problem is formulated which yields the true value of least average causal distortion for any iid source.

1. INTRODUCTION

Finding the value of least average distortion in source data reproduction for the communication scheme below is a well known problem to which classical information theory provides the answer.



Specifically, let R(D) denote the rate distortion function of the source with respect to the appropriate fidelity criterion and C denote the channel capacity. Then the least attainable average distortion, call it D*, is the value that satisfies $R(D^*) = C$. It may only be attainable in a limiting sense. Let $\{(D_i,T_i)\}_{i=1}^{\infty}$ be the average distortions and delay times in source data reproduction for an indexed collection of codes. Usually, if $D_i \to D^*$, then $T_i \to \infty$.

It is of significant practical interest to determine the least average causal distortion (LACD), or more generally the least average T-delay distortion (LATD), in source reproduction. A code will be called a T-delay code if it is physically realizable and produces its estimate $\hat{X}_k$ of the k^{th} source digit X_k before the creation of X_{k+T+1}. A causal code is a 0-delay code.

Others have already remarked upon the value of finding LACD in various situations (e.g. [3], [4]). Causal coding is of interest in team decision theory, wherein the encoder and decoder might be two decision makers. Of extreme interest is the matter of LACD or LATD in the control of large-scale systems, where information about the state of the system may be sent from some locations to others, and stale estimates of the state may be intolerable. (More realistically, this would require a multiterminal setting rather than the simpler single source-channel-destination to which we refer here.) Even in communication situations where delay is not particularly important, the trade-off between less distortion and longer delay is of inherent interest since the complexity/cost of a code usually is linked to its delay time.

Because feedback of previous channel outputs to the encoder is potentially practical in various situations, this is included in the model. Moreover, lower bounds to LACD with feedback will be lower bounds to LACD without feedback.

Two approaches of note to the LACD problem are due to Gorbunov, Pinsker, Petrov et al., [1], [4], [5], and to Teneketzis [2]. Of fundamental interest is the ability of information-theoretic problem formulations to yield the precise value of or lower bounds to LACD for a given source, channel, and fidelity criterion.

Gorbunov et al. and Teneketzis each provide a rate-distortion theoretic problem formulation ($R^0(D)$ and $\overline{D}^0$, resp.) to lower bound LACD. We show, however, that both bounds fail to improve upon the value D^* provided by classical information theory for any independent, identically distributed (iid) source, discrete, memoryless channel, and stationary, per-letter fidelity criterion. Another problem is formulated in the spirit of Gorbunov and Pinsker which is an improvement over their lower bound and Teneketzis's lower bound in that it yields LACD for any iid source.

The $R^0(D)$ problem is a minimization problem that usually is difficult to solve precisely. Since it is used to lower bound LACD, we provide a dual theorem that characterizes $R^0(D)$ as the solution to a variational maximization, thereby more readily permitting the generation of lower bounds to LACD.

2. NOTATION AND ASSUMPTIONS

Notation will be used in accordance with the diagram previously given. $\hat{X}_k$ is the estimate of the k^{th} source digit X_k. X_1^N denotes $(X_1,\ldots,X_N)$, etc.; x_k, y_k, z_k and $\hat{x}_k$ denote generic values of X_k, Y_k, Z_k and $\hat{X}_k$, resp.; $\hat{b}_k$ will also denote a generic value of $\hat{X}_k$ when necessary.

It is assumed a discrete time, discrete-valued stationary source is communicating via a stationary, discrete, memoryless channel. The fidelity criterion is a stationary, per-letter criterion, $d(X_1^N, \hat{X}_1^N) = \frac{1}{N}\sum_{k=1}^{N} d(X_k,\hat{X}_k)$, $I(X_1^N;\hat{X}_1^N)$, $R(D)$ and C have their usual information theory definitions [6]. E denotes expected value, and $A\to B\to C$ means (A,B,C) is Markov in the sense that, given B, A and C are conditionally independent.

3. 0-DELAY D-ENTROPY

Given a source $p(X_1^N)$ and fidelity criterion $d(X_1^N,\hat{X}_1^N)$, Gorbunov and Pinsker [1] define (not using their notation or terminology)

$$R^0(D) = \lim_{N\to\infty} R_N^0(D)$$

where $$R_N^0(D) = \inf I(X_1^N;\hat{X}_1^N)$$

over transition probabilities $q(\hat{X}_1^N|X_1^N)$ such that

(1) $Ed(X_1^N,\hat{X}_1^N) \leqslant D$

(2) $X_{k+1}^N \to X_1^k \to \hat{X}_1^k$, $k=1,\ldots,N$.

Given a channel with capacity C, let D^0 be that value of D satisfying $R^0(D) = C$.

It is helpful to notice that, if constraint 2) is ignored, the definition of the classical rate distortion function is recovered. The Markov conditions of constraint 2) are necessary conditions for causal coding in light of the fact that in constructing $\hat{X}_1^k$, the values of X_1^k are known while the values of X_{k+1}^N are unknown.

In [4] it is stated D^0 is a lower bound to LACD. This is not surprising when it is realized we could equivalently let

$$D^0(R) = \lim_{N\to\infty} D_N^0(R)$$

where $D_N^0(R) = \inf Ed(X_1^N, \hat{X}_1^N)$

over transition probabilities $q(\hat{X}_1^N | X_1^N)$ such that

1) $I(X_1^N; \hat{X}_1^N) \leqslant N\cdot R$

2) $X_{k+1}^N \to X_1^k \to \hat{X}_1^k, k=1,\ldots,N$;

whereupon $D^0 = D^0(C)$. We find ourselves minimizing average distortion over a set of codes which contains all causal codes: from distortion-rate theory, constraint 1) effectively restricts us to the class of codes with arbitrary (but finite) delays, while constraint 2) involves a further restriction as formerly indicated. Hence $D^0(C) \leqslant$ LACD

4. A DUAL THEOREM

In [4] and [5] more results on $R^0(D)$ and LACD are given, primarily for Gaussian sources and mean-squared error, which add to the appeal of D^0 as a meaningful lower bound to LACD. $R^0(D)$ is a minimization problem whose precise solution is usually unobtainable, and since D^0 lower bounds LACD, an alternate maximization formulation for $R_N^0(D)$ would more easily permit the generation of lower bounds to LACD. Because $R_N^0(D)$ is a convex program, this is possible. In Theorem 1, $p(X_1^N)$ is the given source density.

Theorem 1: $R_N^0(D) = \frac{1}{N} \cdot (\sup sD + \sum_{X_1^N} p(x_1^N) \log \lambda(x_1^N))$

over s, $\lambda(\cdot)$ such that

1) $s \leqslant 0$

2) each $\lambda(x_1^N) \geqslant 0$

3) for some collection of numbers $\{\omega(X_1^k, X_1^N), k=1,\ldots,N\}$ and for any $\hat{x}_1^N$

$$\sum_{X_1^N} p(x_1^N)\, \lambda(x_1^N)\, e^{sd(x_1^N,\hat{x}_1^N) + \Omega(x_1^N,\hat{x}_1^N)} \leqslant 1,$$

$$\text{where } \Omega(x_1^N,\hat{x}_1^N)=\sum_{k=1}^{N}\left[\frac{p(x_1^k)}{p(x_1^N)}\,\omega(\hat{x}_1^k,x_1^N)-\sum_{b_{k+1}^N:x_{k+1}^N}\omega(\hat{x}_1^k,x_1^k,b_{k+1}^N)\right].$$

Proof: The proof can be carried out using a generalization of the Lagrange multiplier approach to the dual theorem for $R_N(D)$ as in [6]. Due to its length, the reader is referred to [7] for the details.

Notice if we pick $\omega(\cdot,\cdot)\equiv 0$, then $\Omega(\cdot,\cdot)\equiv 0$ and the dual theorem for $R_N(D)$ is recovered. This emphasizes the fact that $R_N^0(D) \geqslant R_N(D)$, and since both functions are convex and decreasing, that $D^0 \geqslant D^*$ for a given channel. A matter deserving more attention is the possible derivation of tight yet simple lower bounds to $R_N^0(D)$. One indication that this may be difficult is the preceding comment relating $R_N^0(D)$ and $R_N(D)$.

5. A COMPARISON OF D^0 AND D^* FOR IID SOURCES

The quality of D^0 as a lower bound to LACD is naturally of interest. It is very difficult to say anything in general regarding sources with memory. However, the following simple proposition shows $D^0=D^*$ for any iid source, meaning 0-delay D-entropy fails to improve upon the lower bound to LACD provided by classical information theory for any iid source.

Proposition 1. For any iid source the $R^0(D)$ and $R(D)$ curves are identical. Hence also $D^0=D^*$ for any given channel and iid source.

Proof: Let D be given. It is well known that $R(D)=R_1(D)$ for an iid source [6], or equivalently that for any N an iid density $p(X_1^N,\hat{X}_1^N)=\prod_{k=1}^{N} p(\hat{X}_k|X_k)p(X_k)$, where $\prod_{k=1}^{N} p(X_k)$ is the source density, solves the $R_N(D)$ problem. Since it solves the $R_N(D)$ problem, this density satisfies $Ed(X_1^N,\hat{X}_1^N) \leqslant D$ and clearly satisfies $X_{k+1}^N \to X_1^k \to \hat{X}_1^k$ for $k=1,\ldots,N$, so is feasible for the $R_N^0(D)$ problem. Hence $R_N(D) \geqslant R_N^0(D)$. Combining this with the previous observation $R_N(D) \leqslant R_N^0(D)$ implies $R_N^0(D) = R_N(D) = R(D)$. Letting $N\to\infty$ establishes the equality $R^0(D) = R(D)$. Given a channel with capacity C, D^0 is obtained by solving $R^0(D) = C$, and D^* by solving $R(D) = C$. Then $R^0(D) = R(D)$ for any D implies $D^0 = D^*$. ||

6. A LOWER BOUND YIELDING LACD FOR IID SOURCES

While one of the lessons of information theory is that, when considering least average distortion without a delay constraint, the only relevant characteristic of the channel is its capacity C, this is not normally true when considering LACD. With this in mind, given a particular source $\overline{p}(X_1^N)$, channel $\overline{p}(Z_k|Y_k)$ (overbars denote given densities in this section), and fidelity criterion, define

$$\tilde{D} = \inf_N \tilde{D}_N$$

where $\tilde{D}_N = \inf Ed(X_1^N, \hat{X}_1^N)$

over probability densities $p(X_1^N, Y_1^N, Z_1^N, \hat{X}_1^N)$ such that

1) for any x_1^N, $\sum\limits_{y_1^N, z_1^N, \hat{x}_1^N} p(x_1^N, y_1^N, z_1^N, \hat{x}_1^N) = \bar{p}(x_1^N)$

2) for any k and $x_1^N, y_1^k, z_1^k, \hat{x}_1^{k-1}$, $p(z_k | x_1^N, y_1^k, z_1^{k-1}, \hat{x}_1^{k-1}) = \bar{p}(z_k | y_k)$

3) $Z_{k+1}^N \to Z_1^k \to \hat{X}_1^k$ $\quad k=1,\ldots,N$

4) $X_{k+1}^N \to X_1^k \to Y_1^k, Z_1^k$ $\quad k=1,\ldots,N$

5) $X_1^k, Y_1^k \to Z_1^k \to \hat{X}_1^k$ $\quad k=1,\ldots,N$

6) $X_1^k \to Y_1^k \to Z_1^k$ $\quad k=1,\ldots,N$.

This definition is truly in the spirit of the $R^0(D)$ problem in that Markov conditions on the X_1^N, Y_1^N, Z_1^N and $\hat{X}_1^N$ are included which causal codes must obey, namely constraints 3) and 4). Similarly, the Markov conditions in 5) and 6) give necessary conditions that the source be causally connected to the destination through the channel. Constraints 1) and 2) require any feasible density $p(X_1^N, Y_1^N, Z_1^N, \hat{X}_1^N)$ to give back the source and channel probabilities. It is important to notice a consequence of this is that the inequality $\frac{1}{N} I(X_1^N; \hat{X}_1^N) \leq C$, which classical information theory says must hold for any (causal or non-causal) code, is automatically satisfied, by definition of channel capacity. Thus this inequality may be ignored, as may the actual Gorbunov-Pinsker Markov conditions since they, too, can be shown to be implied by a subset of the constraints of the $\tilde{D}$ problem.

$\tilde{D}$ lower bounds LACD merely by definition, because one is minimizing average distortion over a class of codes which contains the class of causal codes. In fact, for any iid source $\tilde{D}$ = LACD, as indicated by the following propositions and remarks.

Proposition 2. For any channel and iid source, $\tilde{D} = \tilde{D}_1$.

Proof: See [7].

Proposition 3. For any channel $\bar{p}(Z|Y)$ and iid source $\bar{p}(X)$, $\tilde{D}_1 = \inf Ed(X, \hat{X})$, where the minimization is over densities $p(X,Y,Z,\hat{X})$ factoring as $p(\hat{X}|Z)\ \bar{p}(Z|Y) p(Y|X)\ \bar{p}(X)$.

Proof: For the $\tilde{D}_1$ problem, constraints 3) and 4) are inoperative. Constraints 5) and 6) reduce to $X,Y \to Z \to \hat{X}$ and $X \to Y \to Z$, resp., which combined with 1) and 2) easily yields the proposition. ||

For an iid source, it can be seen from the formulation for $\tilde{D}_1$ in Proposition 3 that each feasible density for the $\tilde{D}_1$ problem induces a genuinely (and possibly random) causal code. Combining this with Proposition 2 and the fact that $\tilde{D} \leqslant \text{LACD}$, we conclude that, for an iid source, $\tilde{D} = \tilde{D}_1 = \text{LACD}$. An unsurprising and analogous conclusion to be drawn is that the feedback loop and any sort of memory is useless when coding iid sources.

7. ANOTHER LOWER BOUND TO LACD

Given a source $p(X_1^N)$, fidelity criterion, and channel with capacity C, Teneketzis [2] shows $\bar{D}_N^0$ is a lower bound for the N-shot LACD problem. $\bar{D}_N^0$ is defined as

$$\bar{D}_N^0 = \frac{1}{N} \sum_{k=1}^{N} \bar{D}_{N,k}^0$$

where $\bar{D}_{N,k}^0 = \inf Ed(X_k, \hat{X}_k)$ and the minimization is over transition probabilities $q(\hat{X}_k | X_1^k)$ such that $I(X_k; \hat{X}_k | X_1^{k-1}) \leqslant C$. The following proposition and corollary establish the fact that for any iid source $\bar{D}^0 \equiv \inf_N \bar{D}_N^0$ is equal to $\bar{D}_N^0$, for any N and to D*. Thus this new problem offers no improvement over classical information theory in lower bounding LACD for iid sources.

Proposition 4. If the source $p(X_1^N)$ is iid, then $\bar{D}_{N,k}^0 = \inf Ed(X_k, \hat{X}_k)$, where the minimization is over transition probabilities $q(\hat{X}_k | X_k)$ such that $I(X_k; \hat{X}_k) \leqslant C$. Consequently, $\bar{D}_{N,k}^0 = \bar{D}_1^0$ for any N and k, so also $\bar{D}_N^0 = \bar{D}_1^0$.

Proof: Given any $q(\hat{X}_k | X_1^k)$ feasible for $\bar{D}_{N,k}^0$, define

$$\tilde{q}(\hat{X}_k | X_k) = \sum_{X_1^{k-1}} p(x_1^{k-1})\, p(X_k)\, q(\hat{X}_k\; x_1^{k-1}, X_k).$$

To prove the proposition it suffices to show $\tilde{q}(\hat{X}_k | X_k)$ is feasible for $\bar{D}_{N,k}^0$. As it is evident $E_{\tilde{q}} d(X_k, \hat{X}_k) = E_q d(X_k, \hat{X}_k)$, what remains to be shown is that $I_{\tilde{q}}(X_k; \hat{X}_k) \leqslant C$, which is true since $I_q(X_k; \hat{X}_k | X_1^{k-1}) \leqslant C$ and $I_{\tilde{q}}(X_k; \hat{X}_k) - I_q(X_k; \hat{X}_k | X_1^{k-1}) \leqslant 0$ as follows:

$$\begin{aligned} & I_{\tilde{q}}(X_k; \hat{X}_k) - I_q(X_k; \hat{X}_k | X_1^{k-1}) \\ &= \sum_{X_k, \hat{X}_k} p(x_k) \tilde{q}(\hat{x}_k | x_k) \log\left(\frac{\tilde{q}(\hat{x}_k | x_k)}{\tilde{q}(\hat{x}_k)}\right) - \sum_{X_1^k, \hat{X}_k} p(x_1^k) q(\hat{x}_k | x_1^k) \log\left(\frac{q(\hat{x}_k | x_1^k)}{q(\hat{x}_k | x_1^{k-1})}\right) \\ &= \sum_{X_1^k, \hat{X}_k} p(x_1^k) q(\hat{x}_k | x_1^k) \log\left(\frac{\tilde{q}(\hat{x}_k | x_k)}{\tilde{q}(\hat{x}_k)}\right) - \sum_{X_1^k, \hat{X}_k} p(x_1^k) q(\hat{x}_k | x_1^k) \log\left(\frac{q(\hat{x}_k | x_1^k)}{q(\hat{x}_k | x_1^{k-1})}\right) \end{aligned}$$

$$= \sum_{x_1^k, \hat{x}_k} p(x_1^k)\, q(\hat{x}_k|x_1^k) \log\left(\frac{\tilde{q}(\hat{x}_k|x_k)\; q(\hat{x}_k|x_1^{k-1})}{\tilde{q}(\hat{x}_k)\; q(\hat{x}_k|x_1^{k-1})}\right)$$

$$\leq \sum_{x_1^k, \hat{x}_k} p(x_1^k) q(\hat{x}_k|x_1^k) \left[\frac{\tilde{q}(\hat{x}_k|x_k) q(\hat{x}_k|x_1^{k-1})}{\tilde{q}(\hat{x}_k) q(\hat{x}_k|x_1^k)} - 1\right] \quad \text{using } \log t \leq t-1$$

$$= -1 + \sum_{x_1^{k-1}, \hat{x}_k} p(x_1^{k-1}) \frac{q(\hat{x}_k|x_1^{k-1})}{\tilde{q}(\hat{x}_k)} \sum_{x_k} p(x_k)\, \tilde{q}(\hat{x}_k|x_k)$$

$$= -1 + \sum_{x_1^{k-1}, \hat{x}_k} p(x_1^{k-1}) \frac{q(\hat{x}_k|x_1^{k-1})}{\tilde{q}(\hat{x}_k)}\, \tilde{q}(\hat{x}_k) \quad = \quad -1 + 1 = 0. \; ||$$

Corollary to Proposition 4: For any channel and iid source, $\overline{D}^0 = D^*$.

Proof: Since, for any N, $\overline{D}^0 = \overline{D}_N^0 = \overline{D}_1^0$ by Proposition 4, it suffices to show $\overline{D}_1^0 = D^*$. Observe the $\overline{D}_1^0$ formulation is precisely the same as that of the classical distortion rate function D(R), with R = C, for the same source; i.e., $\overline{D}_1^0 = D(C)$. But it is well known [6, p. 24] that D* = D(C).||

8. CLOSING COMMENTS

One purpose of this paper has been to investigate further the nature of two lower bounds to LACD. While these have been shown to offer no improvement over the lower bound already provided by classical information theory for iid sources, this is suspected to be a worst-case analysis; it is likely that they generally offer an improvement upon D* for sources with memory, but this is difficult to quantify.

An advantage to these information theoretic bounds is that one remains within the structured realm of convex programming, where values are relatively easy to estimate compared to actually calculating the true value of LACD, and where dual programs may be obtained. What remains unknown is the general quality of such lower bounds.

The $\tilde{D}$ problem offers an improvement over the lower bounds to LACD of Gorbunov and Pinsker, and Teneketzis, since it is tight for iid sources. However, this new mathematical program is not convex. Due to its greater complexity, it is questionable whether the problem warrants further investigation, rather than dealing directly with a problem that yields LACD for any source and channel.

The nature of the bounds looked at here presupposes that future source outputs are independent of past source estimates at the destination. This renders these bounds useless in the system control situation where the source sends information about part of the system affected by controls, which are in turn partly based upon what is seen at the destination.

ACKNOWLEDGEMENT

This research was supported in part by the U.S. Department of Energy under Contract DE - AC01 - 80RA50419.

REFERENCES

[1] A. K. Gorbunov and M. S. Pinsker, "Nonanticipatory and prognostic epsilon entropies and message generation rates," Problemy Peredachi Informatsii, Vol. 9, No. 3, 12-21, 1973.

[2] D. Teneketzis, "Communication in decentralized control," Ph.D. dissertation, M.I.T., Cambridge, Mass., 1979.

[3] Y. C. Ho, M. P. Kastner and E. Wong, "Teams, market signaling, and information theory," IEEE Trans. on Automatic Control, Vol. AC-23, No. 2, 305-311, 1978.

[4] B. N. Petrov, R. L. Dobrushin, M. S. Pinsker, G. M. Ulanov, and S. V. Ulyanov, "On some interrelations between the theories of information and control," Problems of Control and Information Theory, Vol. 5, No. 1, 31-38, 1976.

[5] A. K. Gorbunov and M. S. Pinsker, "Prognostic epsilon entropy of a Gaussian message and a Gaussian source," Problemy Peredachi Informatsii, Vol. 10, No. 2, 5-25, 1974.

[6] T. Berger, Rate Distortion Theory: A Mathematical Basis for Data Compression, Prentice Hall, Englewood Cliffs, New Jersey, 1971.

[7] G. Munson, "Causal information transmission with feedback," Ph.D. dissertation, Cornell University, Ithaca, New York, 1981.

SUFFICIENCY, AUTOREGRESSION, AND TIME SERIES MODELING

BRADLEY DICKINSON
Department of Electrical Engineering
and Computer Science
Princeton University, Princeton, NJ 08544

ABSTRACT

We present an alternate motivation for the use of autoregressions as a class of time series models. Previously, the principle of maximum entropy has been used in this regard. Here we take a data compression point of view: the purpose of a model is to parametrize a class of stationary Gaussian information sources in such a way that source strings of length N may be encoded into strings of fixed length (independent of N) in an "information lossless" way. The notion of lossless used here corresponds to statistical sufficiency: we want the conditional distribution of the information string given its encoding to be independent of the model parameters. Hence, given the encoding of a source string, a statistically identical string may be generated with a decoder equipped with its own independent random number generator. The decoder is universal, in that it does not depend on the particular source.

The above scenario leads to autoregressive modeling, because autoregressive models are the only ones admitting a sufficient statistic whose dimension is independent of (i.e. does not grow with) the number of observations available. One problem which seems natural to address, given this problem formulation, is the source identification problem, including the important special subproblem of order determination. Since the sufficient statistic captures all the order-dependent information, containing no parametric dependence at all, it makes good sense to base order selection on the statistic.

Another advantage of the sufficient statistic arises in linear predictive coding applications. Computation of the sufficient statistic is considerably easier than determining the predictor coefficients. Thus, for distributed sensor networks, where remote computation capacity may be at a premium, use of the sufficient statistic may be warranted.

Further research on the synthesis, or decoding, operation is required to assess its complexity and practicality.

This work was supported in part by the National Science Foundation, grant ENG77-28523.

CONVERGENCE OF VECTOR QUANTIZERS WITH APPLICATION TO THE DESIGN OF OPTIMAL QUANTIZERS

EFREN F. ABAYA
Department of Electrical Engineering
University of Texas at Austin
Austin, Texas 78712

and

GARY L. WISE
Departments of Electrical Engineering and Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

Suppose that a sequence of probability distribution functions $\{F_n\}$ converges weakly to a distribution function F. When does the sequence of optimal quantizers for the F_n's converge to an optimal quantizer for F? Sufficient conditions are given to guarantee this convergence for scalar and vector quantizers with a general class of distortion measures. These results are used to examine several aspects of quantization, including the existence of minimum r-th power distortion quantizers and the convergence of a recently proposed algorithm for designing optimal quantizers.

I. INTRODUCTION AND PRELIMINARIES

Suppose that a sequence $\{F_n\}$ of probability distribution functions on $\mathbb{R}^k$ converges weakly to a distribution function F. Under what conditions does the sequence of optimal block quantizers for the F_n's converge to an optimal quantizer for F? This question arises in the design of quantizers for unknown or incompletely specified distributions, where it is necessary to use either an estimate of the true distribution or an empirical distribution based on a set of observations. It may be desirable to know that if the estimated or empirical distributions converge to the true underlying distribution, then the quantizers produced approach optimality. The same question occurs in a recent paper on the design of vector quantizers [5]. In most cases, the algorithms for designing optimal quantizers that have appeared in the literature [1,5,6,8] are based upon some form of Max's conditions [8] which are necessary, but not sufficient, for optimality. As a result, the algorithms may converge to a quantizer which is locally optimal for the target distribution F, but is not globally optimal. If an algorithm were started sufficiently close to a global optimum, this problem would not arise. The following proposal by Linde, Buzo and Gray is intended to insure that this always happens [5]. Consider a sequence of distribution functions $\{F_n\}$ converging weakly to the desired distribution F, with F_1 having a unique locally optimal (and therefore also globally optimal) quantizer. If the F_n are taken to be sufficiently "close," then their respective optimal quantizers might reasonably be expected to be close to each other. So if Q_n, an optimal quantizer for F_n, is used as the starting quantizer for F_{n+1}, the algorithm is likely to converge to a globally optimal quantizer Q_{n+1} for

F_{n+1}, despite the presence of other local minima. Intuitively, one might expect the sequence of quantizers $\{Q_n\}$ produced by this doubly iterative procedure to converge to a globally optimal quantizer Q for the limiting distribution F. In this paper, we will discuss conditions under which this convergence can take place.

First we give a few definitions and establish some notation. An N-level k-dimensional vector quantizer is a mapping $Q : \mathbb{R}^k \to \mathbb{R}^k$ which assigns to the input vector x an output vector Q(x) chosen from a finite set of N distinct vectors $\{y_i : y_i \in \mathbb{R}^k, i = 1,2,\ldots,N\}$. When optimal quantizers are being considered, there is no loss of generality in assuming the "nearest neighbor" assignment rule: Q(x) is that member of the set which is nearest to x in Euclidean norm, with ties being broken in some pre-assigned manner. This rule will be adopted throughout the rest of this paper. Scalar quantization (k=1) is considered to be a special case of vector quantization. Generally, the input X is a random vector taking values in $\mathbb{R}^k$ and having a probability distribution function F. The performance of a quantizer Q is measured by a probabilistic mean distortion measure

$$D = D(Q,F) = \int C_0(\|x - Q(x)\|)dF(x). \tag{1}$$

Here $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^k$ and $C_0(t)$ is a non-negative cost function. For simplicity, we will sometimes write $C(x) = C_0(\|x\|)$. As a side result of this paper, we will show that the minimum of (1) can be achieved for the r-th power distortion $D = E\{\|X - Q(X)\|^r\}$. An optimal quantizer is one which minimizes (1) among the class of N-level k-dimensional quantizers.

Let F_n and F be k-dimensional probability distribution functions. The sequence $\{F_n\}$ is said to converge weakly to F (written $F_n \xrightarrow{w} F$) if $F_n(x) \to F(x)$ at every continuity point x of F. We say that $\{F_n\}$converges setwise to F ($F_n \xrightarrow{s} F$) if

$$\lim \int_B dF_n(x) = \int_B dF(x)$$

for every Borel subset B of $\mathbb{R}^k$. Setwise convergence of $\{F_n\}$ implies weak convergence. The following theorem (adapted from [9, p.232]) pertains to setwise convergence.

Theorem 1.1. Let $F_n \xrightarrow{s} F$, and let $\{f_n\}$ and $\{g_n\}$ be two sequences of real-valued Borel measurable functions which converge pointwise to f and g, respectively. Suppose that $|f_n| \leq g_n$ and that $\lim \int g_n dF_n = \int g\, dF < \infty$. Then $\lim \int f_n dF_n = \int f\, dF$.

If $g_n : \mathbb{R}^k \to \mathbb{R}$ is a sequence of real-valued measurable functions, we say that $\{g_n\}$ is uniformly integrable with respect to the sequence of distribution functions $\{F_n\}$ if $\int |g_n|\, dF_n < \infty$ for all n and

$$\lim_{a\to\infty} \sup_n \int_{\|x\|>a} |g_n(x)|dF_n(x) = 0.$$

A "rectangle" $\overset{k}{\underset{i=1}{X}} [a_i,b_i]$ in k-dimensional space will be called a closed cell. On the real line (k=1) a closed cell is identical to a closed interval.

II. DEVELOPMENT

In this section k and N are fixed positive integers. Unless otherwise stated, X denotes a k-dimensional random vector; F_n and F are distribution functions on $\mathbb{R}^k$. Henceforth, it will be assumed that $C_0(t)$ is non-negative and nondecreasing on the ray $[0,\infty)$. Additional restrictions will be imposed as they are needed.

Under the nearest-neighbor assignment rule, a scalar quantizer Q can be represented by the N-tuple $(y_1,y_2,\ldots,y_N)$ of its output levels indexed in increasing order. With this representation, it is natural to say that a sequence of quantizers $\{Q_n\}$ converges to an N-level quantizer Q if the sequence of N-tuples associated with $\{Q_n\}$ converges to the N-tuple representing Q. This implies that $Q_n(x) \to Q(x)$ at all continuity points of Q. Motivated by this observation, we say that a sequence of vector quantizers $\{Q_n\}$ converges to a vector quantizer Q if $Q_n(x) \to Q(x)$ at all continuity points of Q. This defines the convergence of quantizer sequences. Notice that this definition allows the limit quantizer Q to have fewer than N levels. In any case, it follows from this definition that $C(x-Q_n(x)) \to C(x-Q(x))$ at every continuity point of $C(x-Q(x))$. In fact, if C_0 is continuous, then the above convergence is uniform on compact cells.

For the moment, assume that C_0 is continuous. Consider a sequence $\{F_n\}$ converging weakly to the distribution function F, in such a way that for each finite y in $\mathbb{R}^k$, $C(x-y)$ is uniformly integrable with respect to $\{F_n\}$. We will say more about this assumption in Section 3. Let $\{Q_n\}$be a sequence of N-level quantizers converging to the quantizer Q. Assume that Q has N output vectors $y_1,y_2,\ldots,y_N$. Imagine a hypercube (with sides of unit length) centered on y_i, and denote the midpoints of the 2k faces by z_i^j, j=1, 2,...,(2k). Then

$$C(x - Q_n(x)) \leq \sum_{i=1}^{N} \sum_{j=1}^{2k} C(x - z_i^j) \tag{2}$$

for n sufficiently large. Therefore $\{C(x - Q_n(x))\}$ is uniformly integrable with respect to $\{F_n\}$. Applying Theorem A.1, given in the Appendix, we conclude that $D(Q_n,F_n) \to D(Q,F)$. In particular, if Q' is an N-level quantizer, then $D(Q', F_n) \to D(Q', F)$. Suppose that Q_n above is optimal for F_n. Then $D(Q_n, F_n) \leq D(Q', F_n)$. It follows that $D(Q,F) \leq D(Q',F)$. Thus the limit quantizer Q turns out to be optimal for the limit distribution F. As a matter of fact, Q need not have exactly N levels. If it had less than N levels, it would still be optimal for F among quantizers with N levels or less. In summary, we have the following result.

Theorem 2.1. Assume that $C_0(t)$ is non-negative, nondecreasing on $[0,\infty)$ and continuous. Suppose that $F_n \xrightarrow{w} F$, and that Q_n is an optimal N-point quantizer for F_n. If $C(x-y)$ is uniformly integrable with respect to $\{F_n\}$ for each y in $\mathbb{R}^k$, and if $\{Q_n(x)\}$ converges to some quantizer $Q(x)$ at all continuity points of Q, then Q is optimal for F.

The continuity requirement on $C_0(t)$ can be removed by strengthening the mode of convergence of $\{F_n\}$. For example, we can demand that $F_n \xrightarrow{s} F$. This implies pointwise convergence of $F_n(x)$ to $F(x)$, a stronger condition than weak convergence. However, if F and F_n have densities f and f_n, respectively, and $f_n(x) \to f(x)$ almost everywhere, then the two modes of convergence are equivalent. As usual, we assume that C_0 is non-negative and nondecreasing. Assume further that $C(x-y)$ is uniformly integrable with respect to $\{F_n\}$ for each y in $\mathbb{R}^k$, and that $\{Q_n\}$ converges to an N-level quantizer Q. For a fixed y,

$$\left|\int C(x-y)dF_n(x) - \int C(x-y)dF(x)\right| \leq \left|\int_{\|x\|\leq a} C(x-y)dF_n(x)\right.$$

$$\left.-\int_{\|x\|\leq a} C(x-y)dF(x)\right| + \int_{\|x\|>a} C(x-y)dF_n(x) + \int_{\|x\|>a} C(x-y)dF(x).$$

Applying Theorem 2.1 to the first group of terms on the right-hand side and the uniform integrability requirement to the last two terms, we see that $\lim \int C(x-y)dF_n(x) = \int C(x-y)dF(x) < \infty$. Combining this with (2) gives a dominating function which satisfies the hypotheses of Theorem 1.1. Therefore $D(Q_n,F_n) \to D(Q,F)$. Proceeding as in the proof of the previous theorem, we arrive at the following.

Theorem 2.2. Assume that $C_0(t)$ is non-negative and nondecreasing on $[0,\infty)$. Suppose that $F_n \xrightarrow{s} F$, and that Q_n is an optimal N-level quantizer for F_n. If $C(x-y)$ is uniformly integrable with respect to $\{F_n\}$ for each y in $\mathbb{R}^k$, and if $\{Q_n(x)\}$ converges to some quantizer $Q(x)$ at all continuity points of Q, then Q is optimum for F.

It may be noted in passing that $C(x-y)$ is uniformly integrable with respect to $\{F_n\}$ if (a) the F_n's have uniformly bounded supports, or (b) $C_0(t)$ is bounded and $F_n \xrightarrow{w} F$. The key assumption of uniform integrability in the above theorems is not necessary, but cannot be completely dispensed with as the following example shows. Define

$$F_n(x) = \begin{cases} 0 & x < 0 \\ 1-1/n & 0 \leq x < n \\ 1 & n \leq x \end{cases}$$

$$\lim F_n(x) = F(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases} .$$

If $C(x) = |x|$ (mean absolute distortion), then $C(x-y)$ is not uniformly integrable with respect to $\{F_n\}$. Yet the optimum 1-level quantizer for F_n is its median, $\text{med}(F_n) = 0$, which does converge to the median of F. On the other hand, if $C(x) = x^2$ (mean square distortion), the optimum 1-level quantizer for F_n is the expected value $\int x \, dF_n(x) = 1$. This does not converge to the expected value of F, which is $\int x \, dF = 0$. Of course, $(x-y)^2$ is not uniformly integrable with respect to $\{F_n\}$.

Common to the above theorems is the assumption that the sequence of optimal quantizers $\{Q_n\}$ converges. However, all that we really need is a subsequence of $\{Q_n\}$ which converges (to an optimal quantizer). This can be isolated in the following way. Consider the sequence $\{y_{1,n}\}$ of output levels with smallest magnitudes. If this has a finite limit point, then select a convergent subsequence $\{y_{1,n_i}\}$; otherwise, retain the original sequence. Now take the sequence $\{y_{2,n_i}\}$ of output levels with second smallest magnitudes and select a convergent subsequence $\{y_{2,n_{i_j}}\}$, if possible; otherwise, keep the sequence $\{y_{2,n_i}\}$. Do this for all the output levels in turn. At the end of this procedure, either a convergent subsequence $\{Q_{n_j}\}$ will have been selected, or it will have been determined that none of the sequences $\{y_{i,n}\}$ have finite limit points. We now show that this latter possibility can only arise in the trivial case that $C_0(t)$ is a constant with F-probability 1. Suppose that none of the $\{y_{i,n}\}$, $i=1,2,\dots,N$ have finite limit points. Then

$$\lim_{n\to\infty} C(x-Q_n(x)) = \lim_{t\to\infty} C_0(t).$$

Call the limit on the right $C(\infty)$. If $C(\infty)$ is finite (i.e., C_0 is bounded) then it follows from the Dominated Convergence Theorem [9, p.229] that

$$\lim \int C(x-Q_n(x)) \, dF_n(x) = C(\infty). \tag{3}$$

This implies that every quantizer with N levels or less has distortion $D = C(\infty)$. In particular we have $\int [C(\infty) - C(x)] \, dF(x) = 0$. Since the integrand is non-negative, $C(x) = C_0(\|x\|) = C(\infty)$, a constant, with F-probability 1 [9, p.228]. If $C(\infty) = \infty$ (i.e., C_0 is unbounded) it is just as easy to show that (3) holds, which implies that every quantizer with N levels or less has infinite distortion $D(Q,F)$. But we also have, by virtue of uniform integrability, that

$$\begin{aligned} D(Q,F) &= \int C(x - Q(x)) \, dF(x) \\ &\le \liminf \int C(x - Q_n(x)) \, dF(x) \\ &< \infty . \end{aligned}$$

Since this contradicts (3), it follows that there has to be a convergent subsequence of $\{Q_n\}$. In summary, except when C(x) is a constant, the assumption in the above theorems that $\{Q_n\}$ converges (or has a convergent subsequence) is satisfied.

III. APPLICATIONS

The conditions hypothesized above may arise when an N-level scalar quantizer is to be designed for an unknown scalar distribution F, with $C(x) = |x|^r$, $r > 0$ (r^{th}-power distortion). One approach to this problem is to take n random samples from F and form the empirical distribution $\hat{F}_n(x)$ = (# of samples $\leq x$)/n. By the Glivenko-Cantelli Theorem [10] $\hat{F}_n(t) \to F(t)$ uniformly in t almost surely as $n \to \infty$. To apply the theorem in Section 2, we have to show that $|x-y|^r$ is absolutely integrable with respect to $\{\hat{F}_n\}$. By Kolmogorov's Strong Law of Large Numbers [7, p.239]

$$\lim \int_{|x|>a} |x-y|^r \, d\hat{F}_n(x) = \int_{|x|>a} |x-y|^r \, dF(x) \quad \text{almost surely if}$$

$\int |x|^r \, dF(x) < \infty$. Let $\varepsilon > 0$ be given. For any sample sequence for which the above limit holds we may choose a so that the right hand side is less than $\varepsilon/2$, and n_0, depending on the sample sequence, so that

$$\left| \int_{|x|>a} |x-y|^r \, d\hat{F}_n(x) - \int_{|x|>a} |x-y|^r \, dF(x) \right| < \varepsilon/2$$

whenever $n \geq n_0$. Then

$$\int_{|x|>a} |x-y|^r \, d\hat{F}_n(x) < \varepsilon \tag{4}$$

whenever $n \geq n_0$. It therefore follows that $|x-y|^r$ is uniformly integrable with respect to $\{\hat{F}_n(x)\}$ almost surely provided that $\int |x|^r \, dF(x) < \infty$.

It is relatively straightforward to show that an optimal quantizer exists for a distribution with compact support (such as an empirical distribution). Since a quantizer Q is specified by its output levels $(y_1, y_2, \ldots, y_N)$, the distortion D(Q,F) may be considered a function of these levels. For continuous cost functions, this mapping is continuous. Without loss of generality we can restrict $y_1, y_2, \ldots, y_N$ to the convex hull of the support of F. From these considerations, it follows that there is some set of levels $(y_1, y_2, \ldots, y_N)$ which minimizes D(Q,F), and this set describes an optimal N-level quantizer.

If Q_n is an optimal N-level quantizer for the empirical distribution $\hat{F}_n$, then as discussed in Section 2, there is a subsequence of $\{Q_n\}$ which converges to an optimal N-level quantizer for F. Thus we have managed to demonstrate, by sequential arguments, that optimal N-level quantizers exist for general scalar distributions, whenever (a) $C(x) = |x|^r$ and (b) $\int |x|^r \, dF(x) < \infty$. This result can be extended to continuous cost functions for which C(x) =

$O(|x|^r)$ for some positive r and $\int |x|^r \, dF(x) < \infty$. This type of argument has been used previously to show the existence of minimum mean-square error quantizers for the Laplace density [3].

Proceeding to k-dimensional vector quantizers, denote the scalar components of a vector x by $(x^1, x^2, \ldots, x^k)$. Similarly the marginal distributions of a probability distribution F on $\mathbb{R}^k$ will be denoted $F^1, F^2, \ldots, F^k$. Let $I = \mathop{\mathsf{X}}_{i=1}^{k} [-a,a]$ be a closed cell. Using an r^{th}-power distortion measure, we have

$$\int_{I^c} C(x-y)dF(x) = \int_{I^c} \Big[\sum_{i=1}^{k} (x^i-y^i)^2\Big]^{r/2} dF(x)$$

$$\leq k^{r/2} \sum_{i=1}^{k} \int_{|x^i|>a} |x^i-y^i|^r \, dF^i(x^i). \tag{5}$$

From (4) and (5) we see that multi-dimensional empirical distribution functions also result in almost surely uniformly integrable sequences, provided that the r^{th}-moments of the marginal distributions exist or, equivalently, that $\int \|x\|^r \, dF(x) < \infty$. The succeeding developments in the scalar case generalize readily to several dimensions, including the Glivenko-Cantelli Theorem [2].

Another situation in which sequences of distributions arise is in the design of optimal vector quantizers using variational or fixed-point algorithms. Earlier we discussed an algorithm designed to circumvent the problem of convergence to locally optimal solutions. Linde, et al. [5] conjectured that this modified procedure converges to a global optimum, but were unable to prove it rigorously. Theorem 2.1 indicates that the modified algorithm is feasible if the uniform integrability criterion can be satisfied. Moreover, the theorems imply that if two distributions are "close" enough, then their respective optimal output levels are also close to each other. Thus it makes sense to use an optimal quantizer for one distribution as an approximation to the optimal quantizer for the other. The description given above is purposely vague on how close the distributions have to be, since this depends on the particular algorithm used.

Let F denote the distribution of the signal we wish to quantize. A specific choice of F_n recommended by Linde, et al. is the distribution of $W_n = (1-a_n)X + a_n Z$ where X has distribution F, Z has an arbitrary distribution G, and $\{a_n\}$ is a sequence decreasing monotonically to zero. We consider here the r^{th}-power cost function $C_0(t) = t^r$, and assume that

$$E\{\|Z\|^r\} < \infty \ , \ E\{\|X\|^r\} < \infty. \tag{6}$$

It follows from the above that $W_n \to X$ in r^{th}-mean, i.e.

$$\lim_{n\to\infty} E\{\|W_n - X\|^r\} = \lim_{n\to\infty} a_n^r \, E\{\|X-Z\|^r\} = 0.$$

This implies that $\{F_n\}$ converges weakly to F [4, p.452]. Using Minkowski's Inequality or the c_r-inequality it can be shown that

$$\lim E\{\|W_n - p\|^r\} = E\{\|X-p\|^r\} \tag{7}$$

for any vector p in $\mathbb{R}^k$. This is equivalent to saying that $C(x-p) = \|x-p\|^r$ is uniformly integrable with respect to $\{F_n\}$[4, p.138]. All of the conditions of Theorem 2.1 have been verified. Therefore, the sequence $\{Q_n\}$ of optimal quantizers for the F_n's must have a convergent subsequence, and the limit quantizer(s) is optimal for F. Note that except for (6), no other restrictions have been imposed on the distributions F and G. This allows the user considerable latitude in choosing G.

Another suggestion by Linde, et al. [5] is to use the distributions F and G as follows. Take random samples $X_1, X_2, \ldots$ and $Z_1, Z_2, \ldots$ from F and G, respectively, and form the empirical distribution $\hat{F}_n$ of the set of observations $\{(1-a_i)X_i + a_i Z_i : i=1,2,\ldots, n\}$. These may be thought of as independent samples of the random vector W_n. If $x_i = (x_i^1, x_i^2, \ldots, x_i^k)$ and $t = (t^1, t^2, \ldots, t^k)$ are vectors, we will use the notation $x_i \leq t$ as shorthand for the simultaneous set of inequalities $x_i^j \leq t^j$, $j=1,2,\ldots,k$. Then the empirical distribution function $\hat{F}_n$ may be written as

$$\hat{F}_n(t) = \frac{\sum_{i=1}^{n} \chi_t[(1-a_i)X_i + a_i Z_i]}{n} \tag{8}$$

where χ_t is the indicator function $\chi_t[x] = \begin{cases} 1 & x \leq t \\ 0 & \text{else.} \end{cases}$ Observe that $E\{\chi_t[(1-a_i)X + a_i Z]\} = P\{(1-a_i)X + a_i Z \leq t\} \to F(t)$ at all continuity points t of F. Applying the Strong Law of Large Numbers and the Toeplitz Lemma [4, p.89] to (8) yields

$$\lim_{n\to\infty} \hat{F}_n(t) = \lim_{n\to\infty} \frac{\sum_{i=1}^{n} P\{(1-a_i)X + a_i Y \leq t\}}{n} = F(t) \qquad \text{almost surely}$$

for all t in the continuity set of F. Hence $\hat{F}_n$ converges weakly to F, almost surely.

It remains to be shown that $C(x-p) = \|x-p\|^r$ is uniformly integrable with respect to $\{F_n\}$. This is equivalent to showing that [4, p.138]

$$\begin{aligned} &\lim \int \|w-p\|^r \, d\hat{F}_n(w) \\ &= \lim \frac{\sum_{i=1}^{n} \|(1-a_i)X_i + a_i Z_i - p\|^r}{n} \qquad \text{almost surely} \\ &= \int \|x-p\|^r \, dF(x). \end{aligned} \tag{9}$$

Here we will assume that

$$E\{\|X\|^{2r}\},\ E\{\|Z\|^{2r}\} < \infty\ . \tag{10}$$

Then equation (7) holds. Applying the same reasoning as that used to derive (7), we have $\lim E\{\|(1-a_i)X + a_i Z - p\|^{2r}\} = E\{\|X-p\|^{2r}\}$.

This implies that $\sum_{i=1}^{\infty} \frac{E\{\|(1-a_i)X + a_iZ-p\|^r\}}{i^2} < \infty$ and, with (7), that

$$\sum_{i=1}^{\infty} \frac{\mathrm{Var}\{\|(1-a_i)X + a_iZ-p\|^r\}}{i^2} < \infty . \tag{11}$$

This is the condition for a Strong Law of Large Numbers to hold for (9).

Thus we have [7, p.238]
$$\lim \frac{\sum_{i=1}^{n} \|(1-a_i)X_i + a_iZ_i-p\|^r}{n}$$

$$= \lim \frac{\sum_{i=1}^{n} E\|(1-a_i)X + a_iZ-p\|^r}{n} = E\{\|X-p\|^r\}$$

almost surely.
The last line follows from the Toeplitz Lemma. This verifies (9) and therefore the uniform integrability requirement of Theorem 2.1 is satisfied. We conclude that the sequence $\{Q_n\}$ of optimal quantizers for the $\hat{F}_n$'s must have a convergent subsequence, and the limit quantizer is optimal for F. Observe that for this second modification to work, the somewhat stronger condition (10) was assumed.

IV. CONCLUSION

We have established conditions guaranteeing the convergence of a sequence of quantizers. These conditions were then used to establish the existence of optimal r-th power distortion quantizers. Also the convergence of the design techniques proposed by Linde, Buzo, and Gray was established.

APPENDIX

The following convergence theorem is proved.

Theorem A.1. Suppose that $\{F_n\}$ is a sequence of distribution functions on $\mathbb{R}^k$ converging weakly to a distribution function F, and that $g_n : \mathbb{R}^k \to \mathbb{R}$ is a sequence of continuous, real-valued functions converging to a (continuous) function g uniformly on compact cells. Suppose further that $\{g_n\}$ is uniformly integrable with respect to $\{F_n\}$. Then

$$\lim \int g_n dF_n = \int g\, dF.$$

Proof: Let $I = \mathop{\times}_{i=1}^{k} [-a,a]$ be a cell such that F has no discontinuities on the boundary of I. Then

$$\left| \int_I |g_n| dF_n - \int_I |g| dF \right| \le \int_I |g_n-g| dF_n + \left| \int_I |g| dF_n - \int_I |g| dF \right| .$$

The first term on the right hand side goes to zero by uniform convergence, and the second term goes to zero by the Helly-Bray theorem [10, p.83]. Therefore the left-hand side converges to zero, and it follows that

$$\int_I |g| dF \le \liminf \int |g_n| dF_n < \infty .$$

Letting $a \to \infty$, we can invoke Fatou's Lemma [9, p.226] to show that the limit function g is integrable with respect to F. Now we have

$$|\int g_n dF_n - \int g dF| \leq \int_I |g_n - g| dF_n + |\int_I g_n dF_n - \int_I g dF|$$

$$+ \int_{I^c} |g_n| dF_n + \int_{I^c} |g| dF.$$

Letting n and a go to ∞, we get the terms on the right hand side to decrease to zero by virtue of uniform convergence, the Helly-Bray Theorem, uniform integrability, and integrability of g, respectively. This gives the desired result.

ACKNOWLEDGEMENT

This research was supported by the Air Force Office of Scientific Research under Grant AFOSR-81-0047.

REFERENCES

[1] J.D. Bruce, On the Optimum Quantization of Stationary Signals, IEEE Int. Conv. Rec. Pt. I, pp. 118-124, 1964.

[2] L.P. Devroye, A Uniform Bound for the Deviation of Empirical Distribution Functions, J. Multivariate Anal., 7(1977), pp. 594-597.

[3] P. Fleischer, Sufficient Conditions for Achieving Minimum Distortion in a Quantizer, IEEE Int. Conv. Rec. Pt. I, pp. 104-111, 1964.

[4] R.G. Laha and V.K. Rohatgi, Probability Theory, John Wiley and Sons, New York, 1979.

[5] Y. Linde, A. Buzo and R.M. Gray, An Algorithm for Vector Quantizer Design, IEEE Trans. Comm., COM-28 (1980), pp. 84-95.

[6] S.P. Lloyd, Least Squares Quantization in PCM, Bell Telephone Laboratories Paper, Murray Hill, N.J., 1957.

[7] M. Loeve, Probability Theory, D. Van Nostrand Company, Inc., Princeton, 1963.

[8] J. Max, Quantizing for Minimum Distortion, IRE Trans. Inform. Theory, IT-6 (1960), pp. 7-12.

[9] H.L. Royden, Real Analysis, MacMillan Publishing Co., Inc., New York, 1968.

[10] H.G. Tucker, A Graduate Course in Probability, Academic Press, New York, 1967.

k-DIMENSIONAL POLAR QUANTIZERS FOR GAUSSIAN SOURCES

Peter F. Swaszek and John B. Thomas
Dept. of Electrical Engineering and Computer Science
Princeton University
Princeton, N.J. 08544

Abstract

Several investigators have considered polar coordinate quantization of a circularly symmetric source, in particular, the independent bivariate Gaussian source. Their schemes quantize the coordinates independently in an attempt to reduce the MSE below that of an analogous rectangular coordinate quantizer. This paper considers the design of a k-dimensional spherical coordinate quantizer with $k>2$ ($k=2$ matches published results). Examples for the $k=3$ case are presented along with comparisons to the $k=1$ and $k=2$ cases.

I. Introduction

One-dimensional or zero-memory quantization is a process which is widely practiced and has received much theoretical attention. Both optimum uniform and nonuniform quantizers have been designed for a variety of fidelity criteria and source statistics. The canonical example is Max's unit Gaussian probability density function quantizer [1] for the measure Mean Square Error (MSE). The MSE criterion has universal appeal in its tractability and the intuitive notion of noise power (hence signal-to-noise ratio). Rate distortion theory, however, suggests that k-dimensional or block quantizers may be more efficient.

The area of multidimensional quantization began with the works of Huang and Schultheiss [2] and Zador [3]. Huang and Schultheiss considered the problem of quantizing a correlated Gaussian source efficiently. Their solution was to uncorrelate the source by an appropriate linear filter (the inverse of the square root of the covariance matrix of the input) and to quantize the resulting independent Gaussian random variables with separate Max-type quantizers. Hence, they changed the set of basis vectors to improve performance. The quality (number of levels) of each separate quantizer depended upon the variance (or eigenvalue) of each variable. The quantized versions were then recorrelated by the inverse of the original linear device. Zador examined the more general problem of quantizing a k-dimensional source under the assumption of a large number of levels (fine quantization). He employed the compandor model of Bennett on the k-dimensional Euclidean space $\mathbf{R}^k$ and derived error rates depending upon the compressor function and uniform quantizer used.

A major area of interest in multidimensional quantizer design rests in the use of polar coordinates. Specifically, rather than separately quantizing the abscissa and ordinate for $k=2$, a change of variables to polar coordinates is effected. The resulting magnitude and phase are quantized using separate Max-type quantizers. Of particular interest is the quantizing of independent Gaussian random variables. DFT coefficients, holographic data and pairs of inputs from an iid Gaussian source can be considered as the output of a bivariate Gaussian source.

Independent Gaussian variates in rectangular coordinates transform to independent magnitude and phase on the polar coordinate plane. The magnitude is Rayleigh distributed on $[0,\infty)$ and the phase is uniform on $[0,2\pi)$. Minimizing MSE results in a uniform quantizer for the phase angle and a scaled Rayleigh quantizer for the magnitude. Published results indicate that the polar form almost always outperforms the rectangular format for Gaussian variates. Also, the polar form can be used for any circularly symmetric density of which the bivariate Gaussian is one example [4].

This paper concerns an extension of the polar quantizers developed by Pearlman [5] and Bucklew and Gallagher [6] for the MSE criterion. The extension is the treatment of a k-dimensional Gaussian variate with independent elements. Using the k-dimensional transformation to spherical coordinates [7], the resulting magnitude and $k-1$ phase angles will be optimally quantized. The magnitude quantizer will be shown to be a scaled chi (generalized Rayleigh) density quantizer. The first angle is quantized uniformly and the others non-uniformly. Allocation of the number of levels to each separate quantizer will be considered.

The quantizers designed in this paper are all one-dimensional processors (Max-type). Specification entails the computation of the output values and the breakpoints. For a general one-dimensional quantizer $Q(s)$ with N_s-levels operating upon an input s, define $\hat{s}_i$ as the ith output value, $i=1,2,\ldots N_s$, and s_i as the ith breakpoint, $i=1,2,\ldots N_s+1$.

II. k-Dimensional Polar Coordinate Quantizers

The independent, k-dimensional Gaussian density with zero means and unit variances for the random vector $\mathbf{x}$ is of the form:

$$p(\mathbf{x}) = (2\pi)^{-k/2} \exp(-\mathbf{x}^T\mathbf{x}/2) \quad ; \quad \mathbf{x} \in \mathbf{R}^k$$

Following the work of Huang and Schultheiss, each coordinate x_j could be quantized independently by a Max quantizer Q_j. The resulting quantization regions, being the cross product of k intervals, are k-dimensional rectangular parallelepipeds. Each coordinate has a MSE term $E_G(N_j)$ which is the error associated with a unit Gaussian Max quantizer with N_j levels $(N_1 \times N_2 \cdots \times N_k = N)$. The errors, being independent and orthogonal, sum to the total error. It can be shown that this rectangular quantization error is minimized when each $N_j = N^{1/k}$. The total error is then $k \times E_G(N^{1/k})$.

The other obvious set of coordinates (basis vectors) with which to describe an input $\mathbf{x}$ of $\mathbf{R}^k$ is the magnitude and $k-1$ angles of the k-dimensional spherical coordinate system [7]. The following nonlinearities produce these coordinates:

$$r = \left[\sum_{j=1}^{k} x_j^2\right]^{1/2} \quad ; \quad \phi_j = \tan^{-1}\left[x_{j+1} \Big/ \left[\sum_{i=1}^{j} x_i^2\right]^{1/2}\right] \quad , j = 1,2,\ldots k-1$$

The case $k=2$ has already received much theoretical attention and is a special case of this analysis.

The joint density of the spherical Gaussian density is by a change of variables (let $\mathbf{P}^k$ be the spherical analog of $\mathbf{R}^k$):

$$p(r,\underline{\phi}) = f_k(r) \prod_{j=1}^{k-1} f_j(\phi_j) \quad ; \quad [r,\underline{\phi}^T]^T \in \mathbf{P}^k \tag{1}$$

where

$$f_k(r) = \frac{1}{\Gamma(k/2)\, 2^{(k-2)/2}}\, r^{k-1} \exp(-r^2/2) \quad ; \quad r \in [0,\infty)$$

$$f_1(\phi_1) = \frac{1}{2\pi} \quad ; \quad \phi_1 \in [0,2\pi)$$

and

$$f_j(\phi_j) = c_j \cos^j \phi_j \quad ; \quad \phi_j \in [-\pi/2, \pi/2] \; , \; j = 2,3,\dots k-1$$

with the angle coefficients $c_j = 1\times3\times5 \cdots \times(j)/2\times4 \cdots \times(j-1)$ for j odd and $(2\div\pi)\times2\times4 \cdots \times(j)/1\times3\times5 \cdots \times(j-1)$ for j even. The resulting coordinates are statistically independent and will be quantized separately with Max-type quantizers $(Q_r, Q_1, Q_2, \dots Q_{k-1})$.

In $\mathbf{P}^k$, the quantization regions are the intersection of a non-zero width spherical shell centered at zero and a pyramid with apex zero (see Fig. 1 for $k=2$ and 3 examples). The total MSE expression is not as simple as that of the rectangular quantizer and is found below. Another problem of interest is the factorization of the number of levels to each quantizer $(N = N_r \times N_1 \times N_2 \cdots \times N_{k-1})$. In the bivariate case $(k=2)$, it has already been shown [5,6] that a ratio of $N_1 \div N_r \approx 2.6$ minimizes the MSE.

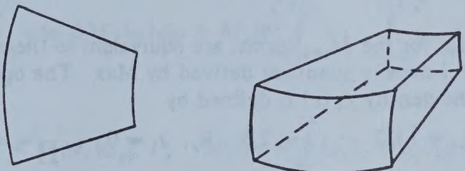


Fig. 1. $k=2$ and $k=3$ polar quantization regions.

For a quantizer with input $\mathbf{x}$ and output $\hat{\mathbf{x}}$ the MSE D is

$$D = \int_{\mathbf{R}^k} |\mathbf{x} - \hat{\mathbf{x}}|^2 \, p(\mathbf{x}) \, d\mathbf{x} \tag{2}$$

Employing the notation $c_i = \cos\phi_i$ and $s_i = \sin\phi_i$, the following transformations yield spherical coordinates:

$$x_1 = r \, c_{k-1} \, c_{k-2} \dots c_2 \, c_1$$

$$x_j = r \, c_{k-1} \, c_{k-2} \dots c_j \, s_{j-1} \quad ; \quad j = 2,3,\dots k-1$$

$$x_k = r \, s_{k-1}$$

Letting $\hat{r}$ and $\hat{\phi}_j$'s represent the quantized magnitude and angles, Eq.(2) reduces to

$$D = \int_{\mathbf{P}^k} [r^2 + \hat{r}^2 - 2r\hat{r} \, \beta(\underline{\phi}, \hat{\underline{\phi}})] \, p(r, \underline{\phi}) \, dr \, d\underline{\phi} \tag{3}$$

where $p(r, \underline{\phi})$ is the spherical Gaussian density in Eq.(1) and

$$\beta(\phi, \hat{\phi}) = s_{k-1}\hat{s}_{k-1} + c_{k-1}\hat{c}_{k-1}[s_{k-2}\hat{s}_{k-2} + c_{k-2}\hat{c}_{k-2}[s_{k-3}\hat{s}_{k-3} + \cdots [s_1\hat{s}_1 + c_1\hat{c}_1]] \cdots]$$

The integral expression for D can be separated. The first two terms in the brackets are independent of $\underline{\phi}$, hence $\underline{\phi}$ can be integrated out of these terms. The last term in the brackets is the product of an integral over r and one over $\underline{\phi}$. Eq.(3) reduces to

$$D = \int_0^\infty r^2 f_k(r) dr + \int_0^\infty \hat{r}^2 f_k(r) dr - 2M_{k-1} \int_0^\infty r\hat{r} f_k(r) dr \tag{4}$$

The term

$$M_{k-1} = \int_0^{2\pi} \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} \beta(\underline{\phi},\underline{\hat{\phi}}) \prod_{i=1}^{k-1} f_i(\phi_i)\, d\underline{\phi}$$

is independent of r and the magnitude quantizer; thus Eq. (4) can be minimized over the choice of the magnitude quantizer. The three integrals present in this expression are all positive since the magnitude r is always positive. Hence, D can also be minimized independently of r by maximizing the value of M_{k-1}.

Magnitude Quantizer

Taking partial derivatives of D in Eq. (4) with respect to the magnitude quantizer ($\partial D/\partial r_i$ and $\partial D/\partial \hat{r}_i$) and locating minimums yield:

$$r_i = (\hat{r}_{i-1} + \hat{r}_i)/2M_{k-1} \quad ; \quad i = 2,3,\ldots N_r, \; r_1 = 0, \; r_{N_r+1} = \infty \tag{5}$$

$$\hat{r}_i = M_{k-1} \int_{r_i}^{r_{i+1}} r f_k(r)\,dr \Big/ \int_{r_i}^{r_{i+1}} f_k(r)\,dr \quad ; \quad i = 1,2,\ldots N_r \tag{6}$$

These expressions, except for the M_{k-1} terms, are equivalent to the equations defining the minimum MSE $f_k(r)$ density quantizer derived by Max. The optimum Max-type, N_r-level quantizer for the density $f_k(r)$ is defined by

$$t_i = (\hat{t}_{i-1} + \hat{t}_i)/2 \quad ; \quad i = 2,3,\ldots N_r, \; t_1 = 0, \; t_{N_r+1} = \infty \tag{7}$$

$$\hat{t}_i = \int_{t_i}^{t_{i+1}} t f_k(t)\,dt \Big/ \int_{t_i}^{t_{i+1}} f_k(t)\,dt \quad ; \quad i = 1,2,\ldots N_r \tag{8}$$

Define a new quantizer with outputs and breakpoints

$$r_i = t_i \text{ and } \hat{r}_i = M_{k-1}\hat{t}_i$$

Substituting into Eqs. (7) and (8), this new quantizer is seen to satisfy the necessary conditions imposed upon the magnitude quantizer by Eqs. (5) and (6). This new quantizer has the output points scaled by the value M_{k-1}. It can be shown that $M_{k-1} \leqslant 1$ (see Appendix) and is usually approximately unity so that the scaling does not remove the output points from their respective regions.

Uniqueness of this quantizer is shown by checking the uniqueness of the unscaled chi quantizer. Applying Fleischer's test [8]:

$$\frac{\partial^2}{\partial r^2} \log f_k(r) < 0$$

to the magnitude density $f_k(r)$, the quantizer is unique for all $k \geqslant 1$.

The magnitude density has variance k. Substituting into Eq. (4), the MSE is

$$D = k - M_{k-1}^2 \sum_{i=1}^{N_r} \left[\int_{r_i}^{r_{i+1}} r f_k(r)\,dr \right]^2 \Big/ \int_{r_i}^{r_{i+1}} f_k(r)\,dr$$

The unscaled chi k quantizer from Eqs. (7) and (8) has MSE:

$$E_{ck}(N_r) = k - \sum_{i=1}^{N_r} \left[\int_{r_i}^{r_{i+1}} r f_k(r)\,dr \right]^2 \Big/ \int_{r_i}^{r_{i+1}} f_k(r)\,dr$$

Hence, the spherical distortion can be written as

$$D = M_{k-1}^2 \; E_{ck}(N_r) + k(1-M_{k-1}^2)$$

Tables of $E_{ck}(N_r)$ and $M_{k-1}(N \div N_r)$ allow the selection of the pair $(N_r, N_\phi = N \div N_r)$ which minimizes D. Since $E_{ck}(N_r) \leqslant k$, picking the largest possible M_{k-1} for a given factorization $(N = N_r \times N_\phi)$ minimizes D.

Maximizing M_{k-1} Over the ϕ_1 Quantizer

Redefine the phase coordinates as $\underline{\omega} = [\phi_2, \phi_3, \ldots, \phi_{k-1}]^T$ and $\theta = \phi_1$. The $\beta(\underline{\phi}, \hat{\underline{\phi}})$ term in Eq. (3) becomes

$$\beta(\underline{\phi}, \hat{\underline{\phi}}) = \gamma(\underline{\omega}) + c_{k-1}\hat{c}_{k-1}c_{k-2}\hat{c}_{k-2} \ldots c_2\hat{c}_2[s_1\hat{s}_1 + c_1\hat{c}_1]$$

where

$$\gamma(\underline{\omega}, \hat{\underline{\omega}}) = s_{k-1}\hat{s}_{k-1} + c_{k-1}\hat{c}_{k-1}[s_{k-2}\hat{s}_{k-2} + c_{k-2}\hat{c}_{k-2}[s_{k-3}\hat{s}_{k-3} + \cdots [s_2\hat{s}_2]] \cdots]$$

is independent of θ. Separating and integrating yields

$$M_{k-1} = \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} \gamma(\underline{\omega}, \hat{\underline{\omega}}) F_2(\underline{\omega}) d\underline{\omega} + M_1(\theta) \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} c_{k-1}\hat{c}_{k-1} \cdots c_2\hat{c}_2 F_2(\underline{\omega}) \, d\underline{\omega} \tag{9}$$

with

$$F_j = \prod_{i=j}^{k-1} f_i(\phi_i) \quad ; \quad M_1(\theta) = \int_0^{2\pi} (s_1\hat{s}_1 + c_1\hat{c}_1) \frac{d\theta}{2\pi}$$

The first integral is independent of θ. The integrand of the second, being non-negative on $\mathop{\mathbf{X}}_{i=2}^{k-1} [-\pi/2, \pi/2]$, insures that the second integral is positive. Hence, maximizing the $M_1(\theta)$ term over θ maximizes M_{k-1}.

Partial derivatives of M_1 with respect to θ_i and $\hat{\theta}_i$ (the quantizer values) yield a uniform quantizer for θ:

$$\theta_i = 2\pi(i-1)/N_1 \quad ; \quad i = 1,2, \ldots N_1+1$$

$$\hat{\theta}_i = \pi(2i-1)/N_1 \quad ; \quad i = 1,2, \ldots N_1$$

The resulting value of M_1 is

$$M_1 = \sin(\pi/N_1) \; / \; (\pi/N_1)$$

These results for $k=2$ are seen to be equivalent to the polar results previously published. Substituting into Eq. (9) yields:

$$M_{k-1} = \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} \delta(\underline{\rho}, \hat{\underline{\rho}}) F_3(\underline{\rho}) d\underline{\rho}$$

$$+ \int_{-\pi/2}^{\pi/2} (s_2\hat{s}_2 + M_1 c_2\hat{c}_2) f_2(\eta) d\eta \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} c_{k-1}\hat{c}_{k-1} \cdots c_3\hat{c}_3 F_3(\underline{\rho}) d\underline{\rho}$$

where $\eta = \phi_2$, $\underline{\rho} = [\phi_3, \ldots \phi_{k-1}]^T$ and

$$\delta(\rho, \hat{\rho}) = s_{k-1}\hat{s}_{k-1} + c_{k-1}\hat{c}_{k-1}[s_{k-2}\hat{s}_{k-2} + c_{k-2}\hat{c}_{k-2}[s_{k-3}\hat{s}_{k-3} + \cdots [s_3\hat{s}_3]] \cdots]$$

Maximizing Over the ϕ_j Quantizer, $j=2,\ldots,k-1$

Sequential maximization, as in the previous section, may be employed to maximize M_{k-1} over the $\underline{\phi}$ quantizers. The general jth iteration is

$$M_{k-1} = \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} \Big(s_{k-1}\hat{s}_{k-1} + c_{k-1}\hat{c}_{k-1}[s_{k-2}\hat{s}_{k-2} + \ldots [s_{j+1}\hat{s}_{j+1}]] \ldots]\Big) F_{j+1}(\underline{\phi})\, d\underline{\phi}$$

$$+ M_j \int_{-\pi/2}^{\pi/2} \cdots \int_{-\pi/2}^{\pi/2} c_{k-1}\hat{c}_{k-1} \cdots c_{j+1}\hat{c}_{j+1} F_{j+1}(\underline{\phi})\, d\underline{\phi}$$

For the same reasons as above, M_{k-1} is maximized by maximizing M_j:

$$M_j = \int_{-\pi/2}^{\pi/2} (s_j \hat{s}_j + M_{j-1} c_j \hat{c}_j) f_j(\phi_j)\, d\phi_j \tag{10}$$

The term M_j is the sum of N_j terms:

$$M_j = c_{j-1} \sum_{i=1}^{N_j} \int_{\phi_i}^{\phi_{i+1}} (\sin\phi \, \sin\hat{\phi}_i + M_{j-1} \cos\phi \, \cos\hat{\phi}_i) \cos^{j-1}\phi \, d\phi \tag{11}$$

Partial derivatives of M_j with respect to the phase quantizer Q_j's parameters ϕ_i and $\hat{\phi}_i$ yield the following necessary conditions:

$$\phi_i = \tan^{-1}\Big[M_{j-1} (\cos\hat{\phi}_i - \cos\hat{\phi}_{i-1}) / (\sin\hat{\phi}_{i-1} - \sin\hat{\phi}_i) \Big] \quad : \quad i = 2,3,\ldots N_j \tag{12}$$

$$\hat{\phi}_i = \tan^{-1}\left[\int_{\phi_i}^{\phi_{i+1}} \cos^{j-1}\phi \, \sin\phi \, d\phi \Big/ M_{j-1} \int_{\phi_i}^{\phi_{i+1}} \cos^{j-1}\phi \, d\phi \right] \quad ; \quad i = 1,2,\ldots N_j \tag{13}$$

The overall result is that in order to minimize the MSE for a polar, k-dimensional quantizer, a factorization of the total number of levels N must be selected: $N = N_r \times N_1 \cdots \times N_{k-1}$. The N_r-level chi k density quantizer is found. The ϕ_1 quantizer is uniform with N_1 levels. Then the $M_j, j=2,\ldots,k-1$ are maximized, each maximization in turn specifying the ϕ_j quantizer Q_j by Eqs. (12) and (13).

III. Iterative Technique

In the discussion of the magnitude and phase quantizers the quantizer parameter vectors **s** and **ŝ** appeared in two sets of conditions:

$$s_i = g(\hat{s}_{i-1}, \hat{s}_i) \text{ and } \hat{s}_i = h(s_i, s_{i+1})$$

If the breakpoints **s** are fixed, the second expression is necessary and sufficient to minimize MSE. Likewise, for fixed outputs **ŝ**, the first expression is necessary and sufficient.

A design algorithm previously proposed [8] is to select a set **ŝ**. Employ $s_i = g(\hat{\mathbf{s}})$ to select the s_i optimally. This set of **s** and **ŝ** has distortion D. Usually, the $\hat{s}_i$ do not satisfy the second equation, so redefining **ŝ** will decrease D. Similarly, now $\mathbf{s} \neq g(\hat{\mathbf{s}})$, so redefining **s** again will further decrease D. This iterative scheme converges to a local minimum of D (when iterating for the phase quantizers the measure M_j will increase successively).

The sequence of the iterative measure D or M_j is seen to approach lower and upper limits respectively. This is due to the fact that D is lower bounded by zero (being the integral of a positive quantity) and M_j is bounded above by unity (see Appendix).

Fleischer [8] notes that the iterative technique considered herein converges slowly, especially for $N>16$. However, for the k-dimensional quantizers the effective N's are the N_j which are small from the factorization.

IV. Examples

k=2: The level factorization is $N=N_r \times N_1$ and $M_{k-1}=M_1=(\sin(\pi/N_1))\div(\pi/N_1)$. The phase quantizer is uniform with N_1 levels and the magnitude quantizer is a scaled Rayleigh (chi 2) quantizer. This matches published results.

k=3: The factorization is $N=N_r \times N_1 \times N_2$. Tables of $E_{c3}(N_r)$ and $M_{k-1}(N_1,N_2)$ were generated. The best combinations for the $k=3$ spherical quantizers are listed in Table 1 along with their Signal-to-Noise Ratio (SNR) rates. The per dimension SNR is defined as

$$SNR = \frac{1}{k} 10 \log_{10} \frac{\sigma^2}{MSE} \quad dB$$

For comparison, the one-dimensional (Max) and two-dimensional (Pearlman) results are also tabulated. The values of N considered correspond to the integers 8 thru 18 cubed which allows easy comparison to the published one and two-dimensional results. The values in parentheses are the actual factorizations employed and in all cases $N_r \times N_1 \times N_2 \leqslant N$. This occurs since cubes of integers do not often have suitable factorizations or some of the results for smaller N are better. From the generated data, the best ratio of levels in each quantizer can be estimated. These are $N_1 \div N_2 \approx 1.9$ and $N_1 \div N_r \approx 2.9$ which result in, for an N-level quantizer, the factorization:

$$N_r \approx .61\, N^{1/3} \quad , \quad N_1 \approx 1.8\, N^{1/3} \quad , \quad N_2 \approx .92\, N^{1/3}$$

N	Rect. SNR	Polar SNR	Three-dim SNR	Best k
512	14.62	14.58	14.52 (5x14x7)	1
729	15.55	15.59	15.58 (5x16x9)	2
1000	16.40	16.31	16.41 (6x18x9)	3
1331	17.16	17.24	17.27 (7x19x10)	3
1728	17.87	17.90	17.93 (7x22x11)	3
2197	18.52	18.63	18.58 (9x22x11)	2
2744	19.13	19.22	19.22 (8x26x13)	3
3375	19.69	19.88	19.87 (9x25x15)	2
4096	20.22	20.35	20.39(10x27x15)	3
4913	20.72	20.80	20.85(10x30x16)	3
5832	21.20	21.34	21.37(11x33x16)	3

Table 1. Comparison of SNR for $k=1,2$ and 3 polar quantizers.

V. Conclusions

The results presented for the three-dimensional polar quantizers show that block encoding of independent Gaussian data (at high rates) is more efficient in a MSE sense that one-dimensional quantizing. This is similar to the result that $k=2$ polar quantizers outperform Max quantizers for $N \geqslant 100$ (>3.3 bits per dimension). The same is expected to hold for larger k. Unfortunately, the computations of M_{k-1} become more involved.

A straightforward and perhaps important extension of this work is the design of k-dimensional polar quantizers with uniform step-size. The solution set is then only the factorization of N and the magnitude quantizer step-size ($k+1$ parameters). This extension is important due to the simplicity of its implementation.

A difficulty noticed in the $k=3$ examples is finding the best factorization of N. Prime values or those having few factorizations usually result in large MSE. Pearlman noted this problem for the $k=2$ case. Hence, the examples may not truly show the advantages of $k=3$ polar quantizing since integers cubed may not factor well.

Appendix

The term M_j, defined in Eqs. (10) and (11), is bounded as follows:

1- Given that $M_{j-1} \epsilon [0,1]$, then $M_j \leqslant 1$.

Eq. (10) expresses M_j as the expectation over $\phi \epsilon [-\pi/2, \pi/2]$ of the function

$$g(\phi) = \sin\phi \sin\hat{\phi} + M_{j-1} \cos\phi \cos\hat{\phi}$$

where $\hat{\phi}$ is the quantized value of ϕ. Hence, upper bounding $g(\phi)$ by unity also bounds M_j by unity. The maximum of $g(\phi)$ is $g_{\max} = \sin\hat{\phi}/\sin\phi$ which is attained at the point $\tan\hat{\phi} = M_{j-1}\tan\phi$. Since $M_{j-1} \epsilon [0,1]$:

$$\tan\hat{\phi} \leqslant \tan\phi \rightarrow |\hat{\phi}| \leqslant |\phi| \rightarrow |\sin\hat{\phi}| \leqslant |\sin\phi| \rightarrow g_{\max} \leqslant 1 \rightarrow M_j \leqslant 1$$

2- Given that $M_{j-1} \epsilon [0,1]$ and the ϕ_j quantizer satisfies Eqs. (12) and (13), then $M_j \geqslant 0$.

The expansion of M_j in Eq. (11) involves cosine integrals and sine-cosine integrals. Rearranging Eq. (13) allows substitution of the sine-cosine integrals over ϕ in Eq. (11). The result is:

$$M_j = c_{j-1} M_{j-1} \sum_{i=1}^{N_i} \int_{\phi_i}^{\phi_{i+1}} (\sin^2\hat{\phi}_i \sec\hat{\phi}_i + \cos\hat{\phi}_i \cos\phi) \cos^{j-1}\phi \, d\phi$$

Since all of the terms present are non-negative over the range of ϕ, $M_j \geqslant 0$.

3- $0 \leqslant M_j \leqslant 1$; $j = 2,3,\ldots k-1$.

Since $M_1 = \sin(\pi/N_1) \div (\pi/N_1)$ and $N_1 \geqslant 1$, then $M_1 \epsilon [0,1]$. This fact, combined with 1 and 2 inductively, shows that $0 \leqslant M_j \leqslant 1$.

Acknowledgement

This research is supported by the Office of Naval Research under Contract N00014-81-K-0146 and by the National Science Foundation under Grant ECS-79-18915.

REFERENCES

[1] J. Max, "Quantizing for Minimum Distortion," *IRE Trans. Inform. Theory,* Vol. IT-6, March 1960, pp.7-12.

[2] J.J.Y Huang and P.M. Schultheiss, "Block Quantization of Correlated Gaussian Random Variables," *IEEE Trans. Comm. Sys.,* Vol. CS-11, Sept. 1963, pp. 289-296.

[3] P. Zador, *Development and Evaluation of Procedures for Quantizing Multivariate Distributions,* Stanford Univ. Dissert., Dept. of Stat., Dec. 1963.

[4] J.A. Bucklew and N.C. Gallagher Jr., "Two-Dimensional Quantization of Bivariate Circularly Symmetric Densities," *IEEE Trans. Inform. Theory,* Vol. IT-25, Nov. 1979, pp.667-671.

[5] W.A. Pearlman, "Polar Quantization of a Complex Gaussian Random Variable," *IEEE Trans. Comm.,* Vol. COM-27, June 1979, pp.892-899.

[6] J.A. Bucklew and N.C. Gallagher Jr., "Quantization Schemes for Bivariate Gaussian Random Variables," *IEEE Trans. Inform. Theory,* Vol. IT-25, Sept. 1979, pp.537-543.

[7] M. Kendall, *The Geometry of n Dimensions,* Griffin Co. Ltd., London, 1962.

[8] P. Fleischer, "Sufficient Conditions for Achieving Minimum Distortion in a Quantizer," *IEEE Int'l. Conv. Rec.,* 1964, part 1, pp.104-111.

SOME RECENT RESULTS ON VECTOR QUANTIZING

N. J. A. SLOANE
Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT

This talk, which is based on joint work with J. H. Conway, will describe recent results on the design of block (or vector) quantizers for uniformly distributed inputs. Particular attention will be paid to the case where the input has constant energy, i.e., is uniformly distributed over a sphere in n-dimensional space. A new measure of distortion is proposed for such quantizers, and a number of quantizers in different dimensions are compared.

Some Results on the Median Filtering of Signals and Additive White Noise†

T. A Nodes and N. C. Gallagher, Jr.

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

Abstract

The first order distribution of the output of a median filter when filtering a known signal plus additive white noise has been derived and is presented along with some examples. In addition, two programs have been written to aid in the design of median filters for the additive white impulse noise case and some of these results are tabulated.

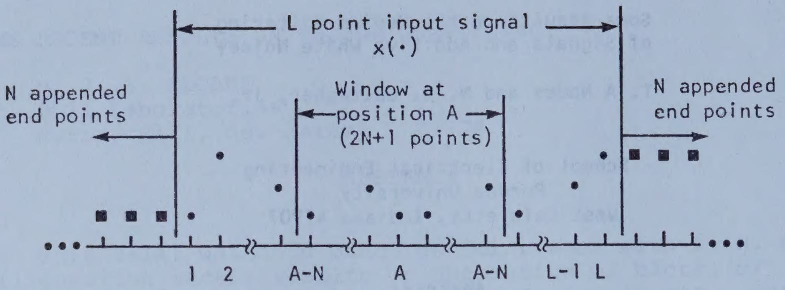
I. Introduction

Median filtering, a method of signal processing which is easily implemented on a digital computer, has been used with success in many applications. These applications include picture processing and speech processing[1,2,3,4] where it is employed to smooth the signal. Previous work in developing the properties of the median filter has been limited to the filtering of deterministic[5] and white noise[8] (i.i.d.) signals. Unfortunately, due to the nonlinearity of the median process, the analysis of the important signal plus additive noise case is not a direct extension of these simpler cases. In this paper, we present some results on the filtering of signals plus additive white noise. Specifically, we have derived the first order output distribution for an arbitrary given signal and noise distribution. This along with several examples is presented in the second part of the paper. In addition, we present some results on the effects of additive impulse noise on median filtered signals. First, however, a review of the standard median filter is in order.

Median filtering is a discrete time process in which a 2N+1 points wide window is stepped across an input signal (see Fig. 1). At each step, the points inside the window are ranked according to their values, and the median value (mid-point) of the ranked set is taken as the output value of the filter for each window position. At both ends of the signal, N end points are appended to allow the filter to reach the edges of the signal. The value of the front endpoints is equal to the value of the first point of the signal, and the value of the rear endpoints is equal to last point of the signal. As an example of this process, consider Fig. 2. Here, a binary signal of length eleven (the ■'s represent the appended endpoints) is median filtered by three different window widths N = 1 (2N+1=3), N = 2 (2N+1=5), and N = 3 (2N+1=7). Notice, for the N=1 case, the signal is unperturbed, while for the N=2 and N=3 cases, the amount of structure in the signal is reduced. A number of signal structures which can be used to define the properties of median filters can now be defined.

---

†The authors gratefully acknowledge the support of the Air Force Office of Scientific Research under grant AFOSR 783605.


L point input signal
x(·)
N appended
end points
Window at
position A
(2N+1 points)
N appended
end points
1
2
A-N
A
A-N
L-1
L

The output of the median filter, $Y(A)$ is given by

$Y(A)$ = the median value of $\{x(a-N),\ldots,x(A-1),x(A),x(A+1),\ldots,x(A+N)\}$

Fig.1: The Median Filter

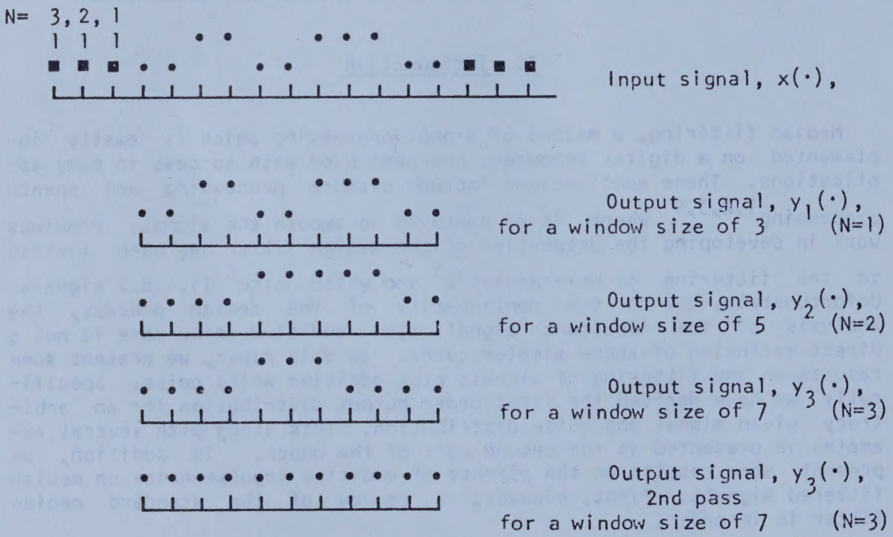

N= 3, 2, 1
1 1 1
Input signal, x(·),
Output signal, y₁(·),
for a window size of 3 (N=1)
Output signal, y₂(·),
for a window size of 5 (N=2)
Output signal, y₃(·),
for a window size of 7 (N=3)
Output signal, y₃(·),
2nd pass
for a window size of 7 (N=3)

Fig.2: Effects of window size on a median filtered signal

A constant neighborhood is a region of at least N+1 consecutive points all of which are identically valued.

An edge is a monotonically rising or falling set of points surrounded on both sides by constant neighborhoods.

An impulse is a set of N or less points whose values are different from the surrounding regions and whose surrounding regions are identically valued constant neighborhoods.

A root is a signal which is not modified by filtering.

Gallagher and Wise [5,6] have shown that, while impulses are eliminated by median filtering, constant neighborhoods and edges are unperturbed, and in fact, only signals composed solely of constant neighborhoods and edges are roots to the median filter. Again referring to Fig. 2, note that the signal is a root of the N=1 median filter but not for filters with N greater than one. However, after one pass of the N=2 filter or two passes of the N=3 filter the resulting outputs are roots of their respective filters. In fact, Gallagher and Wise have also proven that any signal of length L is reduced to its root after at most $\frac{1}{2}\cdot(L-2)$ successive passes by any median filter. Furthermore, any root of a median filter with a particular window size is also a root of any median filter with a smaller window size.

For i.i.d. (white) random signals, Kulman and Wise[8] have derived the second order statistics of the median filter. They further show that for all the distributions which they have investigated, which include most of the common ones, the median filter has a low pass effect on the signal spectrum, and thus increases the correlation. In fact, this is also often true with more general signals; however, due to the nonlinear nature of the filter there are cases where the second moment bandwidth of a signal is actually increased upon median filtering and thus the correlation decreased. Thus, one must use some care in applying the low pass assumption to median filters.

II. Output Distribution

Section I reviewed much of the previous work on properties of median filtered deterministic and i.i.d. signals. As stated earlier, the more general case of filtering signals plus additive noise is much more difficult to analyze. In this section, the first order distribution of the output of a median filter with a known signal and additive white noise input is given. This is used in program Dis to compute some statistics of the output of the median filter several examples of which are given.

If the output of median filter at position m has a distribution of $F_Y(q,m)$ and the input a distribution of $F_X(q,i) = F_{noise}(q - s_i)$ where s_i = signal at position i, then the output distribution is

$$F_y(q,m)=\sum_{k=1}^{N+m-k}\cdots\sum_{a_k=a_{k-1}+1}^{N+m}\left\{\left[\prod_{i\varepsilon a_1,\ldots,a_k}[1-F_x(q,i)]\right]\times\left[\prod_{i\varepsilon\overline{a_1,\ldots,a_k}}(F_x(q,i))\right]\right\}+\prod_{i=m-N}^{m+N}F_x(q,i)$$

where

$2\cdot N + 1$ = window width

$$\overline{\{a_1,a_2,\cdots,a_k\}} \cup \{a_1,a_2,\cdots,a_k\} = \{1,2,\cdots,(2\cdot N+1)\}$$

$$\sum_{i=a}^{b} f(i) = \begin{cases} f(a) + \cdots + f(b) & \text{if } a \leq b \\ 1 & \text{if } a > b \end{cases}$$

This result comes about from combining all possible combinations of the points inside the window such that at least N+1 of them have values $\leq q$. It is straightforward to extend this result to obtain the first order output distribution for any arbitrary input (any arbitrary random process) if the (2N+1)th order distribution is known at every position, however, this result is somewhat cumbersome and is not presented here.

The above equation was incorporated into program Dis to compute the value of the first order median filter output distribution, $F_y(q,m)$, for a signal plus white noise input. This is then used to numerically evaluate some of the statistical properties of the output at each position m. Specifically, Dis computes the value of the mean, $E\{Y\}$, the standard deviation, σ_Y, the mean square error, M.S.E. $(= E\{(y_i - s_i)^2\})$, and the absolute error, A.E. $(= E\{|y_i - s_i|\})$ at every position. These terms may then be plotted as in Fig. 4 through Fig. 7 or averaged over the signal and tabulated as in Tables 1 and 2. These examples illustrate some of the effects of median filtering signals plus noise. Two different distributions (impulsive and gaussian) both with the same two noise power levels are used in these examples. The impulse noise used is double sided symmetric with heights of ± 3 and probabilities $P^+ = P^- = 0.001$ and 0.05 for noise powers of $\sigma_n^2 = 0.018$ and 0.90 respectively. Likewise, the gaussian noise powers are also $\sigma_n^2 = 0.018$ and 0.90. For comparison, results are also given for windowed averaging filters.

First, consider a constant signal. The results from a constant signal indicate the effects that the noise distribution by itself has on the filter output. The results for several such cases using averaging and median filters are given in Table 1. It can be seen that the Average Filter does somewhat better than the median filter when filtering gaussian noise. This is expected since for a set window width the Average Filter is the optimum M.S.E. estimator in this case. However, when impulse noise is present the median filter reduces the output noise power by orders of magnitude more than the Average filter. This is due to the ability of the median filter to totally eliminate low probability high power impulses which is not possible with linear systems. In fact, it can be shown that for a fixed window width the median filter is the optimum MAP estimator in this case. In general, for constant signals median filters have been found to out-perform averaging type filters when the tails of the additive noise density are extensive[9] compared to the gaussian case. Also certain types of general signals are particu-

Table 1: Mean square error of median and average filter outputs with constant signal plus noise inputs. Window width=2N+1

Input Additive Noise	Average Filter n=1	Average Filter n=3	Average Filter n=5	Median Filter n=1	Median Filter n=3	Median Filter n=5
Impulse						
$\sigma_{in}^2=0.018$	6.000E-3	2.573E-3	1.637E-3	5.396E-5	6.285E-10	8.2612E-15
$\sigma_{in}^2=0.90$	3.000E-1	1.286E-1	8.182E-2	1.305E-1	3.484E-3	1.044E-4
Gaussian						
$\sigma_{in}^2=0.018$	6.000E-3	2.573E-3	1.637E-3	8.909E-3	4.610E-3	3.215E-3
$\sigma_{in}^2=0.900$	3.000E-1	1.286E-1	8.182E-2	4.046E-1	1.902E-1	1.243E-1

larly suitable for median filtering irregardless of the noise distribution.

As pointed out earlier, many systems generate signals which are not amenable to the general spectrum separation techniques that ease the design of linear filters. Often this is due to the presence of sharp edges in an otherwise low frequency signal. Such structures tend to be roots to median filters making the median filter a good alternative for smoothing such signals. One such signal is used here to illustrate the effects of median filtering these signals when additive white noise is present. This signal ranges from -2 to 2 and consists of edges and constant neighborhoods. Figures 3 through 6 plot the filter output expected value and standard deviation ($E\{Y_i\}$, $E\{Y_i\} + \sigma_{Y_i}$ and $E\{Y_i\} - \sigma_{Y_i}$) at each position as solid lines and the original uncorrupted signal as a dashed line. For comparison, the results for a windowed average filter are shown in Fig. 3. As with the median filters, the window width = 2 • N + 1.

As illustrated above, the median filter does an excellent job of eliminating impulses (see also Table 3). However, with non-constant signal structures, other types of errors become prevalent when impulse noise is present. Foremost among these is edge jitter. This effect is present even at low noise levels and is not reduced by using larger windows as illustrated in Fig. 4a and Fig. 5a. This effect will be further discussed in Section III. Fig. 5 also shows the effects of filtering with larger windows. The final peak of the signal is only five points wide instead of the six (= N + 1) necessary to pass through an N=5 median filter unperturbed. Fig. 5 also illustrates another error form which occurs when the width of a plateau or valley approaches N+1 points. One or two impulses of the correct sign located within such a plateau will cause the whole plateau to drop to the closest point below it, which can be a substantial change.

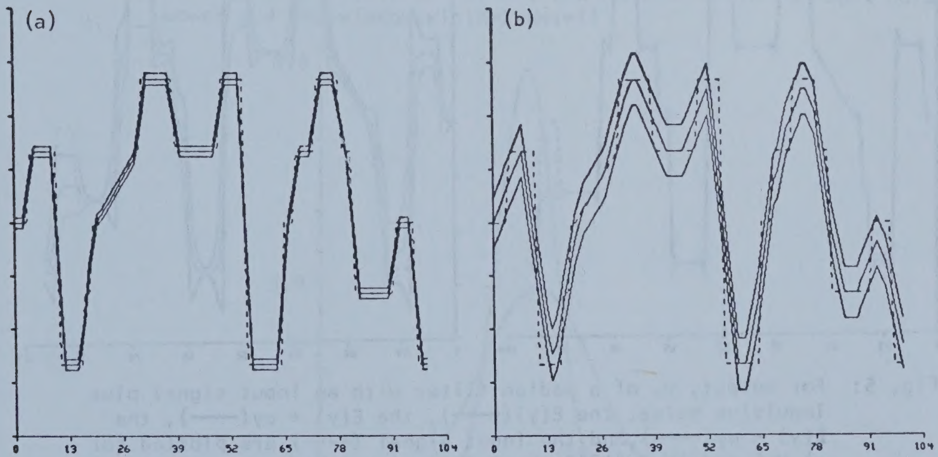


Fig. 3: For output, y, of an averaging filter, with input signal plus noise the E{y}(——), the E{y} + σy(——), the E{y} - σy(——), and the input signal (----) are plotted for a) N=1 and PN=0.018 and b) N=3 and PN=0.90 where PN=input noise power and the window width=2 N+1

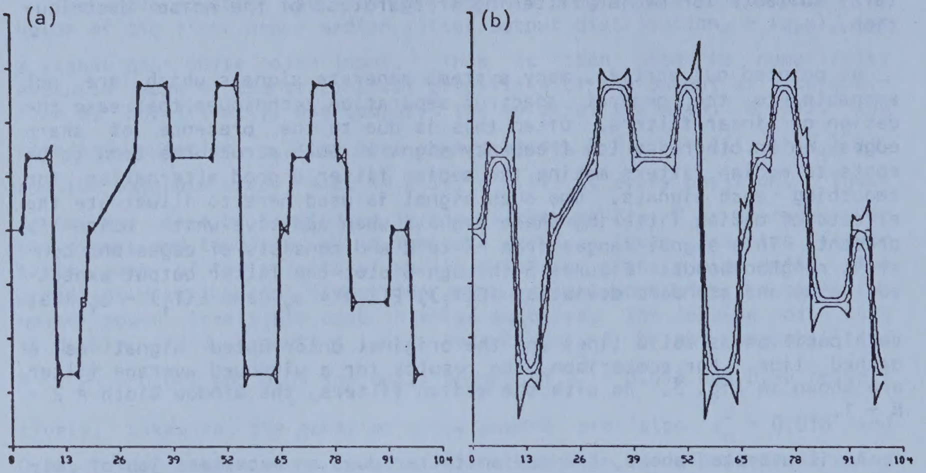


Fig. 4: For output, y, of a median filter with an input signal plus impulsive noise, the E{y}(——), the E{y} + σy(——), the E{y} - σy(——), and the input signal (----) are plotted for a) N=3 and PN=0.018(P+=P-=0.001 and Height(Imp.)=±3) and b) N=3 and PN=0.90(P+=P-=0.05 and Height(Imp.)=±3) where PN=input noise power and the window width=2·N+1=7

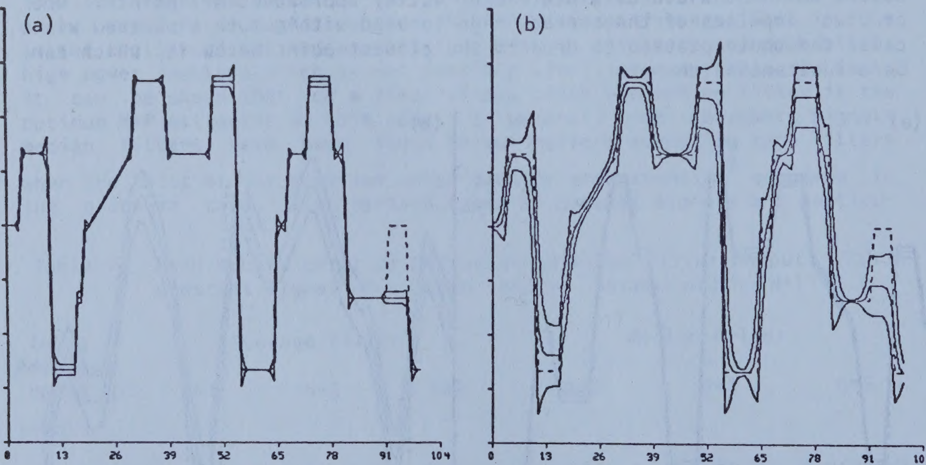


Fig. 5: For output, y, of a median filter with an input signal plus impulsive noise, the E{y}(——), the E{y} + σy(——), the E{y} - σy(——),and the input signal (----) are plotted for a) N=5 and PN=0.018(P+=P-=0.001 and Height(Imp.)=±3) and b) N=5 and PN=0.90(P+=P-=0.05 and Height(Imp.)=±3) where PN=input noise power and the window width=2·N+1=11

Conversely, when gaussian noise is present, quite different results are obtained. As can be seen from Fig. 6, in this case the Std. Dev. of the output is much more smooth and constant, than with impulse noise, and the plots more closely resemble the results of the Average Filter much more closely than before. This is further illustrated in Fig. 7 which plots the density of the output of the N=3 median filter at position 34 (as reviewed in Fig. 6). Notice that while it is shifted and the Std. Dev. reduced, it is still fairly smooth, symmetrical, and bell shaped (although the tails do exhibit some assymmetry which is unobserv-

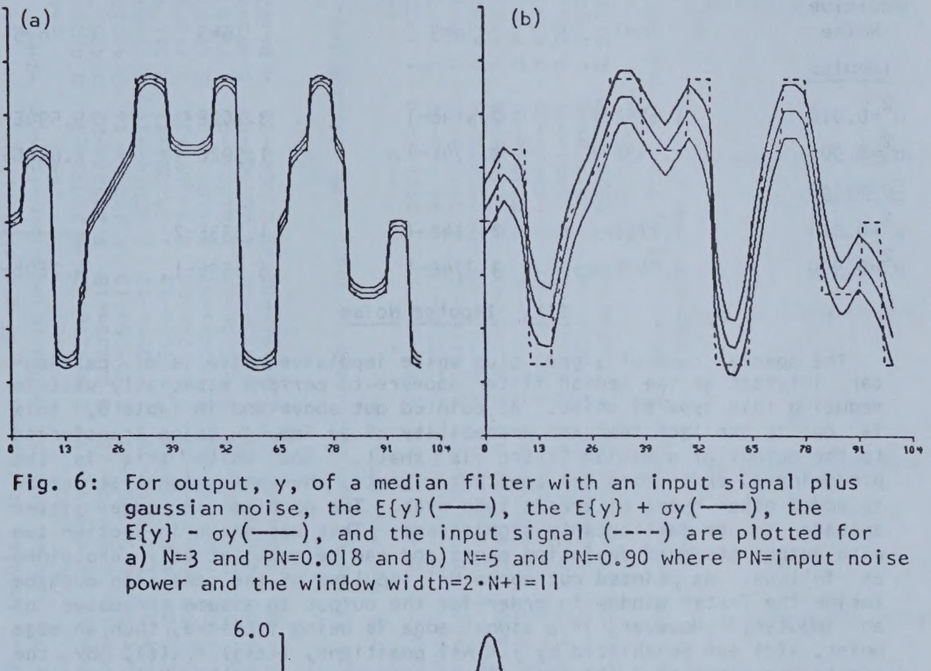


Fig. 6: For output, y, of a median filter with an input signal plus gaussian noise, the $E\{y\}$(——), the $E\{y\} + \sigma y$(——), the $E\{y\} - \sigma y$(——), and the input signal (----) are plotted for a) N=3 and PN=0.018 and b) N=5 and PN=0.90 where PN=input noise power and the window width=2·N+1=11

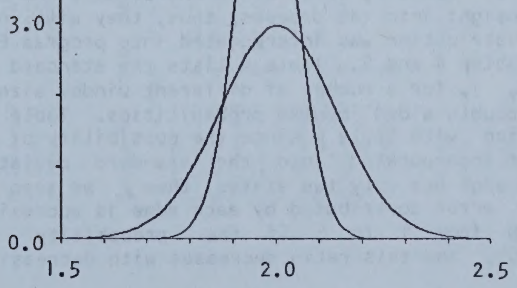


Fig. 7: The output density (upper curve) of an N=3 median filter with an input of signal plus gaussian noise (density: lower curve) with PN=0.018 at position 34 (see Fig. 6)

able in Fig. 7). This is due to the fact that gaussian noise perturbs almost all the input points by a small amount rather than just a few by a large amount as is the case with impulse noise. However, the median filter tracks the signal more closely than the Average filter does. A summary of the average M.S.E.s for the above filter is given in Table 2.

Table 2: Average mean square error, $\overline{\text{M.S.E.}}$, of filter output filtering a 100 point signal and additive noise (see Fig. 3 through Fig. 7)

Input Additive Noise	Average Filt.		Median Filt.	
	n=1	n=3	n=3	n=5
Impulse				
$\sigma^2=0.018$	1.125E-1	2.514E-1	3.202E-3	5.590E-2
$\sigma^2=0.90$	4.043E-1	3.774E-1	1.502E-1	2.640E-1
Gaussian				
$\sigma^2=0.018$	1.125E-1	2.514E-1	1.333E-2	———
$\sigma^2=0.900$	4.043E-1	3.774E-1	3.553E-1	4.280E-1

III. Impulse Noise

The special case of signal plus white impulsive noise is of particular interest as the median filter appears to perform especially well in reducing this type of noise. As pointed out above and in Table 3, this is due to the fact that the probability of an impulse being transferred to the output of a median filter is small. And while this is the predominate error form for constant signals, when more signal structure is added other types of errors take over. The problem of edge jitter appears to be particularly significant. This was shown in Section two with output standard deviation plots and can be qualitatively explained as follows. As pointed out above N+1 impulses of the same sign must be inside the filter window in order for the output to assume the value of an impulse. However, if a signal edge is being filtered, then an edge point, x(t) can be shifted by $j < N+1$ positions, $Y(t+j) = x(t)$, by the simple presence of j impulses of the correct signs within N+1 positions of t. Narrow ($\sim$ N+1 positions wide) plateaus and valleys are also susceptible to impulses; however, these structures are much less common than edges in most signals.

The distribution of edge jitter, $j(y(t+j) = x(t))$ has been derived. The equations, however, are rather untractable and do not lead to any particular insight into the process; thus, they will not be presented here. The distribution was incorporated into program Edg which was used to compile Tables 4 and 5. Table 4 lists the standard deviation of the edge jitter, j, for a number of different window sizes (window width = 2N + 1) and double sided impulse probabilities. Table 3 should be used in conjunction with Table 4 since the possibility of an impulse at the output is not incorporated into the standard deviation computation. Note if the edge has only two states, then , as seen in Table 4, the mean square error contributed by each edge is approximately doubled by increasing N from 1 to 5 if the probability of impulse is $P_+ = P_- = 0.05$, and this ratio decreases with decreasing $P_+ = P_-$. This

Table 3: Prob. of an impulse at the output of a median filter

p+=p-= Prob. of + or - impulse at the input vs. Prob. of impulse at the output with wind. =2n+1

p+=p-	n=1	n=2	n=3	n=4	n=5	n=6
1.00e-05	5.99990e-10	1.99994e-14	6.99969e-19	2.51985e-23	9.23933e-28	3.43167e-32
1.00e-04	5.99900e-08	1.99940e-11	6.99692e-15	2.51853e-18	9.23327e-22	3.42900e-25
1.00e-03	5.99000e-06	1.99401e-08	6.96928e-11	2.50535e-13	9.17296e-16	3.40212e-18
2.00e-03	2.39200e-05	1.59043e-07	1.11019e-09	7.97054e-12	5.82814e-14	4.31684e-16
5.00e-03	1.48750e-04	2.46282e-06	4.27994e-08	7.64906e-10	1.39222e-11	2.56679e-13
1.00e-02	5.90002e-04	1.94100e-05	6.69956e-07	2.37774e-08	8.59369e-10	3.14599e-11
2.00e-02	2.32007e-03	1.50713e-04	1.02619e-05	7.18246e-07	5.11873e-08	3.69481e-09
5.00e-02	1.37569e-02	2.15378e-03	3.52202e-04	5.91515e-05	1.01136e-05	1.75157e-06
1.00e-01	5.02024e-02	1.48470e-02	4.55957e-03	1.43575e-03	4.60250e-04	1.49544e-04
2.00e-01	1.65473e-01	8.78393e-02	4.80858e-02	2.69196e-02	1.53382e-02	8.86622e-03

Table 4: Std. deviation of edge jitter of median filtered signals with additive Imp. noise

p+=p-= Prob. of + or - impulse vs. Std. Dev. with wind. =2n+1

p+=p-	n=1	n=2	n=3	n=4	n=5	n=6
1.00e-05	6.32446e-03	7.74608e-03	8.94441e-03	1.00002e-02	1.09546e-02	1.18323e-02
1.00e-04	1.99970e-02	2.44986e-02	2.82885e-02	3.16275e-02	3.46462e-02	3.74222e-02
1.00e-03	6.31507e-02	7.75748e-02	8.95770e-02	1.00150e-01	1.09709e-01	1.18499e--01
2.00e-03	8.91745e-02	1.09867e-01	1.26871e-01	1.41846e-01	1.55385e-01	1.67835e-01
5.00e-03	1.40362e-01	1.74446e-01	2.01505e-01	2.25292e-01	2.46795e-01	2.66569e-01
1.00e-02	1.97008e-01	2.48301e-01	2.87106e-01	3.21015e-01	3.51655e-01	3.79831e-01
2.00e-02	2.74400e-01	3.55038e-01	4.12025e-01	4.60268e-01	5.04877e-01	5.45333e-01
5.00e-02	4.14095e-01	5.72890e-01	6.77901e-01	7.61845e-01	8.35481e-01	9.02651e-01
1.00e-01	5.40000e-01	8.12157e-01	1.00338e+00	1.15004e+00	1.27171e+00	1.37858e+00
2.00e-01	6.40000e-01	1.07016e+00	1.43122e+00	1.73486e+00	1.99263e+00	2.21482e+00

Table 5: Amount of jitter with 90% certainity (Prob(jitter>k)<0.1) for Sig. and Imp. noise

p+=p-= prob. of + or - impulse vs. k (jitter limit with 90% certainty) with wind. =2n+1

p+=p-	n=1	n=2	n=3	n=4	n=5	n=6
1.00e-05	0	0	0	0	0	0
1.00e-04	0	0	0	0	0	0
1.00e-03	0	0	0	0	0	0
2.00e-03	0	0	0	0	0	0
5.00e-03	0	0	0	0	0	0
1.00e-02	0	0	0	0	1	1
2.00e-02	0	1	1	1	1	1
5.00e-02	1	1	1	1	1	1
1.00e-01	1	1	2	2	2	2
2.00e-01	impulse	2	3	3	3	4

increase, then, must be reconciled with a corresponding decrease in the probability of an impulse at the output by a factor of 1,300 for the same parameters as above. Further information can be obtained from Table 5 which gives the amount of jitter with 90% certainty. The listing "impulse" in this table indicates that the probability of the output assuming the value of an impulse is greater than 10%. The use of these tables in conjunction with the deterministic properties developed by Gallagher and Wise[11] should greatly facilitate the design of median filters used in filtering signals with additive impulse noise as they help to quantify the various trade offs available in such designs.

IV. Conclusion

The first order distribution of the output of a median filter for a signal plus white noise input was presented. Using this, the statistics of several examples with impulsive and gaussian noise were computed and given. These illustrate some of the properties of median filtering. Edge jitter and narrow plateau jitter are seen to be the dominate error modes for impulse noise. For gaussian additive noise, the output more closely resembles that of an average filter but with a larger standard deviation and closer tracking of edges. For the additive impulse noise case, some statistical properties of the edge jitter is tabulated. These results should aid in the design of median filters since they illustrate many of the properties the designer can expect from these filters in the important signal plus white noise case. However, much more work needs to be done in this area to develop easier to use and more general descriptions of the properties of the filter while retaining some quantitative ability.

References

1. J. W. Tukey, "Nonlinear (Nonsuperposable) Methods for Smoothing Data," in Cong. Rec., 1974 EASCON, p. 673.
2. T. S. Huang, G. T. Yang, and G. Y. Tang, "A Fast Two-Dimensional Median Filtering Algorithm," IEEE Trans. Accoust., Speech, and Signal Processing, vol. ASSP-27, pp. 13-18, Feb. 1979.
3. L. R. Rabiner, M. R. Sambur, and C. E. Schmidt, "Applications of a Nonlinear Smoothing Algorithm to Speech Processing," IEEE Trans. Acoust., Speech, and Signal Processing, Vol. ASSP-23, pp. 552-557, Dec. 1975.
4. N. S. Jayant, "Average and Median Based Smoothing Techniques for Improving Digital Speech Quality in the Presence of Transmission Errors," IEEE Trans. on Commun., Vol. COM-24, pp. 1043-1045, Sept. 1976.
5. N. C. Gallagher, Jr. and G.L. Wise, "Passband and Stepband Properties of Median Filters," IEEE Trans. Accoust. Spch. Sig. Proc., to be published.
6. S-G. Tyan, It has come to our attention that S-G. Tyan has also proved a number of these properties. We have not seen the proofs and can only speculate as to their form.
7. H. A. David, Order Statistics, (1970), Wiley, New York.
8. F. Kuhlmann and G. L. Wise, "On Spectral Characteristic of Median Filtered Independent Data," IEEE Trans. on Commun., Vol. COM-29, No. 9, pp. 1374 Sept. 1981.
9. Aulx F. Velleman, "Definition and Comparison of Robust Nonlinear Data Smoothing Algorithms," Journal of the American Statistical Assoc., pp. 609, Sept. 1980.

AN ADAPTIVE FILTER USING POLYNOMIAL LEAST SQUARES METHODS

G. H. XIA
University of California, Irvine
Irvine, California and
Chinese Academy of Space Technology
Peking, China

ALLEN R. STUBBERUD
University of California, Irvine
Irvine, California

ABSTRACT

An adaptive filter is developed for the situation in which the dynamic equations of a system and the statistical properties of the associated signals are unknown. Using polynomial fitting techniques and least squares methods, a recursive algorithm is devised. The algorithm tracks both the system parameters and the order of the polynomial in real time. The algorithm is efficient in both computation and storage.

INTRODUCTION

In engineering practice, we are often faced with problems in which the dynamics equations are unknown or too complicated to describe, the statistical properties of processes are unknown, and the capability of the control computer is limited. In these cases, normal least squares estimates and Kalman filters may not be efficient for generating estimates of the system state. In this paper an adaptive filter is developed which can be used in these cases. In the recursive algorithms, the system process is represented by a polynomial thus the parameter matrices in the algorithm can be written in an analytical form. The order of the polynomial can be chosen adaptively. The paper presents two algorithms, one for an infinite memory filter (IMF) and one for a finite memory filter (FMF).

INFINITE MEMORY FILTER

A polynomial

$$y_n(i) = \sum_{j=0}^{n} a_j \cdot (ih)^j \qquad i = 1,2,3,\ldots \tag{1}$$

and its derivatives

$$y_n^{(m)}(i) = \sum_{j=m}^{n} \frac{j!}{(j-m)!} a_j \cdot (ih)^{j-m} \qquad \begin{array}{l} i = 1,2,3,\ldots \\ m = 0,1,2,\ldots,n \end{array} \tag{2}$$

are used to fit the process variable of the system. In Equations (1) and (2) h is the constant sampling period and i is the time index. The measurement data y(i), i = 1,2,3,..., is thus related to $y_n(i)$ through the equations

$$y(i) = y_n(i) + v(i) \qquad i = 1,2,3,\ldots \tag{3}$$

$$v(i) = e(i) + n(i) \qquad i = 1,2,3,\ldots \tag{4}$$

where e(i) and n(i) are the truncation error and random error, respectively. Now to determine values for the coefficients a_j, j = 0,1,...,n, a quadratic cost function

$$J = \sum_{i=1}^{k} \left[y(i) - y_n(i) \right]^2 \tag{5}$$

is chosen and minimized.

The condition that minimizes the cost function is

$$\frac{\partial J}{\partial a_j} = 0 \qquad j = 1,2,\ldots,n \tag{6}$$

Then the estimate of the parameter vector, $\hat{A}_n(k)$, can be obtained as

$$\hat{A}_n(k) = M_n^{-1}(k)P_n(k)Y(k) \tag{7}$$

where

$$A_n(k) \triangleq (a_0 \ \ a_1 \ \ a_2 \ \ldots \ a_n)^T \tag{8}$$

$$Y(k) \triangleq [y(1) \ y(2) \ y(3) \ \ldots \ y(k)]^T \tag{9}$$

$$P_n(k) = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ h & 2h & 3h & \ldots & kh \\ \multicolumn{5}{c}{\dotfill} \\ h^n & (2h)^n & (3h)^n & \ldots & (kh)^n \end{bmatrix} \tag{10}$$

and

$$M_n(k) = P_n(k)P_n(k)^T = \begin{bmatrix} \sum_{i=1}^{k} 1 & \sum_{i=1}^{k} ih & \ldots & \sum_{i=1}^{k} (ih)^n \\ \sum_{i=1}^{k} ih & \sum_{i=1}^{k} (ih)^2 & \ldots & \sum_{i=1}^{k} (ih)^{n+1} \\ \multicolumn{4}{c}{\dotfill} \\ \sum_{i=1}^{k} (ih)^n & \sum_{i=1}^{k} (ih)^{n+1} & \ldots & \sum_{i=1}^{k} (ih)^{2n} \end{bmatrix} \tag{11}$$

In the above T denotes a transpose.

Letting

$$\hat{X}_n(k) = [\hat{y}(k) \ \hat{y}(k)' \ \hat{y}(k)'' \ \ldots \ \hat{y}(k)^{(n)}]^T \tag{12}$$

and

$$T_n(k) = \begin{bmatrix} 1 & kh & (kh)^2 & \ldots & (kh)^n \\ 0 & 1 & 2kh & \ldots & n(kh)^{n-1} \\ \multicolumn{5}{c}{\dotfill} \\ 0 & 0 & 0 & \ldots & n! \end{bmatrix} \tag{13}$$

the relationship between $\hat{X}_n(k)$ and $Y(k)$ can be written as

$$\hat{X}_n(k) = T_n(k)\hat{A}_n(k) = T_n(k)M_n^{-1}(k)P_n(k)Y(k). \tag{14}$$

Since

$$P_n(k+1) = [P_n(k)\ \Delta p_n] \tag{15}$$

where

$$\Delta p_n = \left[1\ (k+1)h\ (k+1)^2h^2\ \ldots\ (k+1)^n h^n\right]^T \tag{16}$$

then

$$M_n^{-1}(k+1) = M_n^{-1}(k) - M_n^{-1}(k)\Delta p_n\left[I + \Delta p_n^T M_n^{-1}(k)\Delta p_n\right]^{-1}\Delta p_n^T M_n^{-1}(k) \tag{17}$$

and it is possible to develop a recursive algorithm as

$$\hat{X}_n(k+1) = \Phi_n(k)\hat{X}_n(k) + K_n(k)[y(k+1) - H(k)\Phi_n(k)\hat{X}_n(k)] \tag{18}$$

where

$$\Phi_n(k) = T_n(k+1)T_n^{-1}(k) \tag{19}$$

$$H_n(k) = \Delta p_n^T T_n^{-1}(k+1)$$

$$K_n(k) = fT_n(k+1)M_n^{-1}(k)\Delta p_n \tag{20}$$

$$f = \left(1 + \Delta p_n^T M_n^{-1}(k)\Delta p_n\right)^{-1}. \tag{21}$$

If the initial $\hat{X}_n(0)$ is specified, the estimate $\hat{X}_n(k)$ can be computed recursively.

FINITE MEMORY FILTER

If in the IMF the truncation error e(k) is zero and n(k) is a stationary white noise with zero mean, then the least squares solution has the optimal statistical properties of being consistent, unbiased and efficient. In practice, however, e(k) is usually not equal to zero, so the longer the data length being processed the larger the error will be. If the order of polynomial is increased to reduce the effect of e(k), then the random error n(k) will increase. Therefore, the data length is often kept constant (as new data is obtained, the oldest data is discarded). In this case, it is expedient to write the polynomial approximating the system process as:

$$y_n(k-\bar{k}+i) = \sum_{i=0}^{n} a_j \cdot (ih)^j \qquad i = 1,2,3,\ldots \tag{22}$$

Using the same method as in the previous section, we obtain

$$\hat{X}_n(k) = T_n(\bar{k})M_n^{-1}(\bar{k})P_n(\bar{k})Y(k) \tag{23}$$

where

$$Y(k) = [y(k-\bar{k}+1)\ y(k-\bar{k}+2)\ \ldots\ y(k)]^T \tag{24}$$

Now, we generate the relationship between $\hat{X}_n(k+1)$ and $\hat{X}_n(k)$. From (23), as k → k+1, it is obvious that

$$\hat{X}_n(k+1) = T_n(\bar{k})M_n^{-1}(\bar{k})P_n(\bar{k})Y(k+1). \tag{25}$$

From the definitions of p_n and Y we have

$$P_n(\bar{k})Y(k+1) = \begin{bmatrix} \sum_{i=1}^{k} y(k-\bar{k}+1+i) \\ \sum_{i=1}^{k} y(k-\bar{k}+1+i)ih \\ \cdots\cdots\cdots\cdots \\ \sum_{i=1}^{k} y(k-\bar{k}+1+i)(ih)^n \end{bmatrix}$$

$$= Q_n P_n(\bar{k})Y(k) + \Delta p_n y(k+1) - Dy(k-\bar{k}+1) \tag{26}$$

where

$$Q_n = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -h & 1 & 0 & 0 & \cdots & 0 \\ h^2 & -2h & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ (-h)^n & \binom{n}{1}(-h)^{n-1} & \binom{n}{2}(-h)^{n-2} & \binom{n}{3}(-h)^{n-3} & \cdots & 1 \end{bmatrix} \tag{27}$$

$$\Delta p_n = [1 \quad \bar{k}h \quad (\bar{k}h)^2 \; \ldots \; (\bar{k}h)^n]^T \tag{28}$$

$$D = (1 \quad 0 \quad 0 \; \ldots \; 0)^T \tag{29}$$

Note that $y(k-\bar{k}+1)$ is the final term in (26) thus all measurement data must be stored during the estimation procedure. Fortunately from (22) we see that $y(k-\bar{k}+1)$ is the oldest data in the sequence. Since

$$y(k-\bar{k}+1) = \sum_{j=0}^{n} a_j \cdot (1h)^j + D^T P_n^T(\bar{k}) M_n^{-1}(\bar{k}) P_n(\bar{k}) Y(k) \tag{30}$$

then the measurement data need not be stored, but the estimate must be. Thus the computer storage is substantially reduced. Substituting (30) into (26), it is seen that

$$P_n(\bar{k})Y(k+1) = \left[Q_n - DD^T P_n^T(\bar{k}) M_n^{-1}(\bar{k})\right] P_n(\bar{k})Y(k) + \Delta p_n y(k+1) \tag{31}$$

The recursive formula is obtained by substituting (31) into (25) as follows:

$$\hat{X}_n(k+1) = \bar{\Phi}_n(\bar{k})\hat{X}_n(k) + \bar{K}_n(\bar{k})\left[y(k+1) - \bar{H}_n(\bar{k})\bar{\Phi}_n(\bar{k})\hat{X}_n(k)\right] \tag{32}$$

$$\bar{\Phi}_n(\bar{k}) = T_n(\bar{k}) M_n^{-1}(\bar{k}) Q_n M_n(\bar{k}) T_n^{-1}(\bar{k}) \tag{33}$$

$$\bar{K}_n(\bar{k}) = T_n(\bar{k}) M_n^{-1}(\bar{k}) \Delta p_n \tag{34}$$

$$\bar{K}_n(\bar{k})\bar{H}_n(\bar{k}) = T_n(\bar{k}) M_n^{-1}(\bar{k}) DD^T P_n(\bar{k}) T_n^{-1}(\bar{k}) \tag{35}$$

Thus far we have developed IMF and FMF formulae for a fixed value of n. For a system known to be time-invariant, either the IMF and the FMF can be used but normally the IMF is used for starting until $k=\bar{k}$, then the FMF is used.

RECURSIVE SOLUTION FOR TIME-VARIANT SYSTEMS

Recursive algorithms are of great interest in control and estimation problems. They result not only in computational benefits, but also admit the possibility of proceeding with the gathering of observed data until the specified accuracy of the parameters is achieved. For an unknown time-variant system, it is necessary to identify the order n and the coefficients a_i of the polynomial. Therefore, an on-line recursive algorithm in which n and k are changeable must be developed. First consider a recursive algorithm for n when the sequence of observations Y(k), i=1,2,...,k, is given.

From the definition, as n → n+1, the coefficient matrices can be partitioned as follows:

$$T_{n+1}(k) = \begin{bmatrix} T_n(k) & \Delta T_{n+1} \\ 0 & (n+1)! \end{bmatrix}, \qquad P_{n+1}(k) = \begin{bmatrix} P_n(k) \\ p_{n+1} \end{bmatrix} \tag{36}$$

and

$$M_{n+1}(k) = \begin{bmatrix} M_n(k) & L \\ L^T & \sum_{i=1}^{k} (ih)^{2n+1} \end{bmatrix}. \tag{37}$$

When $k \geq n+1$, $M_{n+1}(k)$ is nonsingular, so the inverse of matrix

$$M_{n+1}^{-1}(k) = \begin{bmatrix} M_n^{-1}(k)+g^{-1}M_n^{-1}(k)LL^TM_n^{-1}(k) & -g^{-1}M_n^{-1}(k)L \\ -g^{-1}L^TM_n^{-1}(k) & g^{-1} \end{bmatrix} \tag{38}$$

exists. The submatrices of the matrices above are given by

$$\Delta T_{n+1} = [(kh)^{n+1} \quad (n+1)(kh)^n \quad \ldots \quad \tfrac{1}{2}(n+1)!(kh)]^T \tag{39}$$

$$p_{n+1} = [h^{n+1} \quad (2h)^{n+1} \quad \ldots \quad (kh)^{n+1}] \tag{40}$$

$$L = \left[\sum_{i=1}^{k} (ih)^{n+1} \quad \sum_{i=1}^{k} (ih)^{n+2} \quad \ldots \quad \sum_{i=1}^{k} (ih)^{2n+1} \right]^T \tag{41}$$

$$g = \sum_{i=1}^{k} (ih)^{2n+1} - L^TM_n^{-1}(k)L \tag{42}$$

Letting n→n+1 in (23) and substituting (36) and (38) into (23) yields

$$\hat{X}_{n+1}(k) = \hat{X}_n(k) + G_{n+1}(k)\hat{y}_e(k) \tag{43}$$

$$\hat{y}_e(k) = (n+1)!g^{-1}\left[p_{n+1}Y(k) - L^TT_n^{-1}(k)\hat{X}_n(k)\right] \tag{44}$$

$$G_{n+1}(k) = \frac{1}{(n+1)!}\left[\Delta T_{n+1} - T_n(k)M_n^{-1}(k)L\right] \tag{45}$$

It is evident that the estimate of the order n+1 is equal to the estimate of the order n corrected by the term proportional to $\hat{y}_e(k)$. Starting from n=1 and iterating the computation of (43) and (44), the estimate of any order can be obtained.

In order to track the variety of the dynamic system properties in real time, the order n and the time index k should be changed. We shall derive an economic and powerful recursive algorithm in the IMF-Case and the FMF-Case, respectively.

(I) IMF-Case. We wish to show how to recursively compute $\hat{X}_{n+1}(k+1)$ as the order n and the time index k increase.

According to (43) and (44), as k → k+1, we have

$$\hat{X}_{n+1}(k+1) = \hat{X}_n(k+1) + G_{n+1}(k+1)\hat{y}_e(k+1) \tag{46}$$

$$\hat{y}_e(k+1) = [(n+1)!]g_1^{-1}\left[p_{n+1}Y(k+1) - L_1^T T_n^{-1}(k+1)\hat{X}_n(k+1)\right] \tag{47}$$

where

$$g_1 = \sum_{i=1}^{k+1} (ih)^{n+1} - L_1^T M_n^{-1}(k+1)L_1 \tag{48}$$

$$L_1 = L + ((k+1)h)^{n+1}\Delta p_n \tag{49}$$

$$p_{n+1} = P_{n+1}Y(k) + ((k+1)h)^{n+1} \tag{50}$$

Combining (47) and (50) with (44) yields

$$\hat{y}_e(k+1) = \phi(k+1,k)\hat{y}_e(k) + k(k)[y(k+1) - h(k)\hat{X}_n(k)] \tag{51}$$

where

$$\phi(k+1,k) = gg_1^{-1} \tag{52}$$

$$k(k) = (n+1)!g_1^{-1}\left[((k+1)h)^{n+1} - L_1^T M_n^{-1}(k)\Delta p_n\right] \tag{53}$$

$$h(k) = \left[((k+1)h)^{n+1} - L_1^T M_n^{-1}(k)\Delta p_n\right]^{-1}\left[((k+1)h)^{n+1} - fM_n^{-1}(k)\Delta p_n\right] \tag{54}$$

An adaptive filtering scheme based on (46), (47) and (18) is shown in Fig. 1. It consists of two parts: an on-line filter and an order detector. If the output of the order detector, $\hat{y}_e(k+1)$, equals zero, then the estimate $\hat{X}_{n+1}(k+1)$ equals $\hat{X}_n(k+1)$, that is, the order is equal to n. In this case, only the filter must be implemented. If $\hat{y}_e(k+1)$ does not approach zero, then the order detector should continue operating until the order is determined.

(II) FMF-Case. Using the same method as above, a recursive algorithm for the FMF is given by

$$\hat{X}_{n+1}(k+1) = \hat{X}_n(k+1) + G_{n+1}(\bar{k})\hat{y}_e(k) \tag{55}$$

$$\hat{X}_n(k+1) = \bar{\Phi}_n(\bar{k})\hat{X}_n(k) + \bar{K}_n(\bar{k})[y(k+1) - \bar{H}_n(\bar{k})\ \bar{\Phi}_n(\bar{k})\hat{X}_n(k)] \tag{56}$$

$$\hat{y}_e(k+1) = \hat{y}_e(k) + c(\bar{k})[y(k+1) - \bar{h}(\bar{k})\hat{X}_n(k)] \tag{57}$$

in which the form of filtering equation is the same as for the IMF, but the coefficient matrices are different. The coefficient matrices in (56) have been given by (33)-(35). The gain matrix in (55) is given by

$$G_{n+1}(\bar{k}) = G_{n+1}(k)\Big|_{k=\bar{k}} \tag{58}$$

Letting (58) equal zero, we can choose an adequate data length $\bar{k}$ that makes the truncation error less even if the order is incorrect. We shall discuss how to determine the optimal n and $\bar{k}$ in the next section.

The coefficients in (57) are

$$c(\bar{k}) = (n+1)!\,g^{-1} \cdot \left((\bar{k}h)^{n+1} - L^T M_n^{-1}(\bar{k})\Delta p_n\right) \tag{59}$$

$$\bar{h}(\bar{k}) = \left((\bar{k}h)^{n+1} - L^T M_n^{-1}(\bar{k})\Delta p_n\right)^{-1} \cdot L^T\left[I + q_{n+1}M_n(\bar{k}) - M_n^{-1}(\bar{k})\left(Q_n - DD^T P_n(\bar{k})M_n^{-1}(\bar{k})\right)\right] T_n^{-1}(\bar{k}) \tag{60}$$

where

$$q_{n+1} = \left[(-h)^{n+1} \quad (-h)^n\binom{n+1}{n} \quad (-h)^n\binom{n+1}{n-1} \quad \cdots \quad (-h)\binom{n+1}{1}\right] \tag{61}$$

$$\binom{n+1}{i} = \frac{(n+1)!}{i!(n+1-i)!} \tag{62}$$

Fig. 2 shows an adaptive filtering scheme for the FMF. It also consists of two parts: an on-line filter and an $(n,\bar{k})$ detector.

DETERMINATION OF n AND $\bar{k}$

Determination of n and $\bar{k}$ is the most important item if the dynamic equations of a system are unknown. Correct n and $\bar{k}$ will result in a high accuracy estimate. If they are incorrect it will cause high frequency random error and low frequency truncation error which may possibly make the filter diverge.

We assume that the real system is of order m, but a polynomial of order n is used to fit it. This will produce a truncation error

$$\Delta X_e = X_m - X_n = G(m-n,k)y_e(k) \tag{63}$$

Here G(m-n,k) and $y_e(k)$ can be obtained by referring to the derivation of equation (55).

We see that there exist three ways to minimize the truncation error in equation (63).

(A) $y_e(k) = 0$. The condition for the vector $y_e(k)$ to be zero is $n \geq m$. This causes the random error to increase.

(B) $G(m-n,\bar{k}) = 0$. Using this equation, a shorter data length is usually obtained. Also, this causes the random error to increase.

(C) Minimize the estimation error including the truncation error and random error. We do this as follows:

We define the estimate error

$$\Delta\hat{X} = X_m - \hat{X}_n = \Delta X_e + \Delta\hat{X}_N \tag{64}$$

where

$$\Delta\hat{X}_N = X_n - \hat{X}_n = -T_n(k)M_n^{-1}(k)P_n(k)N(k) \tag{65}$$

$$N(k) = [n(1) \quad n(2) \quad n(3) \ \ldots \ n(k)]^T \tag{66}$$

The covariance matrix corresponding to the estimation error is

$$R = E[\Delta X \Delta X^T] \tag{67}$$

A criterion that the variance r_{ii}^2 is equal to or less than an allowable value C_i^2 can be used to determine n and $\bar{k}$. That is, if

$$r_{ii} \le C_i \tag{68}$$

is satisfied, then the switch S shown in Fig. 2 is off; otherwise, S is on.

If the error vectors have the following properties

$$E[N(k)N(k)^T] = \sigma^2 I \tag{69}$$

$$E\left[N(k)y_e(k)^T\right] = E\left[y_e(k)N(k)^T\right] = 0 \tag{70}$$

then

$$R = R_e + R_N \tag{71}$$

where

$$R_e = G(m-n,k)E\left[y_e(k)y_e(k)^T\right]G(m-n,k)^T \tag{72}$$

$$R_N = T_n(k)M_n^{-1}(k)T_n^T(k)\sigma^2 \tag{73}$$

To exhibit the characteristic of the estimation error with respect to n and $\bar{k}$, we give the explicit function of the variance for n = 0,1.

$$r_N^2 = \frac{1}{\bar{k}}\sigma^2 \qquad n = 0$$

$$\left\{\begin{aligned} r_N^2(1,1) &= \frac{2(2\bar{k}-1)}{\bar{k}(\bar{k}+1)}\sigma^2 \\ r_N^2(2,2) &= \frac{12h-2}{\bar{k}(\bar{k}+1)(\bar{k}-1)}\sigma^2 \end{aligned}\right. \qquad n = 1$$

$$r_e^2 = 0 \qquad \forall\ n = m \tag{74}$$

$$r_e^2 = \tfrac{1}{4}((\bar{k}-1)h)^2 \cdot y_e^2(k) \qquad n = 0,\ m = 1$$

$$\left\{\begin{aligned} r_e^2(1,1) &= \frac{((\bar{k}-1)(\bar{k}-2)h^2)^2}{144}\, y_e^2(k) \\ r_e^2(2,2) &= \frac{((\bar{k}-1)h)^2}{4}\, y_e^2(k) \end{aligned}\right. \qquad n = 1,\ m = 2$$

According to the results above, we see that for a fixed n, r_e increases (r_N decreases) with increasing $\bar{k}$; for a certain $\bar{k}$, r_e decreases (r_N increases) with increasing n. Therefore, the optimal n and $\bar{k}$ that minimize the sum of the truncation error and random error must exist.

SIMPLIFICATION OF THE COEFFICIENT MATRICES

The formulae have been developed for the general case. All of these formulae are in recursive form, so the computation and storage requirements are significantly reduced, but the computation of the coefficient matrices is still complicated. It is possible to simplify the results further without making approximations.

We see that all of the coefficient matrices, such as $\Phi_n(k)$, $K_n(k)$, $G_n(k)$, $\phi(k+1,k)$, $c(k)$, etc., consist of the power series of natural numbers which can be written in compact forms:

$$\sum_{i=1}^{k} i^0 = k$$

$$\sum_{i=1}^{k} i^1 = \tfrac{1}{2}k(k+1) = c$$

$$\sum_{i=1}^{k} i^2 = \frac{1}{3}(2k+1)c \tag{75}$$

$$\sum_{i=1}^{k} i^3 = c^2$$

Note the third expression in (75) for example. If the left side is calcuated for k = 100, then 99 additions and 100 multiplications are required. If the right side is used for k = 100, however, only one addition and 4 multiplications are required. Therefore, if we use the right sides in (75) and substitute them into the coefficient matrices, then the matrices can be represented in the simple analytic formulae. We give an example as follows:

Assume the process can be described by a 1st order polynomial. According to (13), (16) and (11) we have

$$T_1(k) = \begin{pmatrix} 1 & kh \\ 0 & 1 \end{pmatrix} \qquad T_1(k+1) = \begin{pmatrix} 1 & (k+1)h \\ 0 & 1 \end{pmatrix}$$

$$\Delta p_n = [1 \quad (k+1)h]^T$$

$$M_1^{-1}(k) = \begin{bmatrix} \sum_{i=1}^{k} i^0 & \sum_{i=1}^{k} ih \\ \sum_{i=1}^{k} ih & \sum_{i=1}^{k} (ih)^2 \end{bmatrix}^{-1} = \frac{2}{k(k-1)} \begin{bmatrix} 2k+1 & -3h^{-1} \\ -3h^{-1} & \dfrac{6h^{-2}}{k+1} \end{bmatrix}$$

Substituting the expressions above into (19) and (21) yields

$$\Phi_1(k) = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix} \qquad K_1(k) = \frac{2}{(k+1)(k+2)} \begin{pmatrix} 2(k-1) \\ 3h^{-1} \end{pmatrix}$$

$$H_1(k) = (\ 1 \quad 0\ )$$

Using the method shown above, we can obtain the coefficient matrices of any order in any case. The computation on line is very simple, although the derivation off line is very tedious.

CONCLUSION

In this paper, an adaptive filter using polynomial least squares methods is presented. This algorithm may provide a solution even in cases when other methods have failed. The system process which is unknown can always be fitted by a polynomial. The order of the polynomial and the data length can be chosen to track the various parameters in real time by a so-called $(n,\bar{k})$ detector. The on-line computation is very simple since the coefficient matrices are given by simple analytical functions of $\bar{k}$. Prediction and smoothing problems are easily generalized from the algorithm developed in this paper.

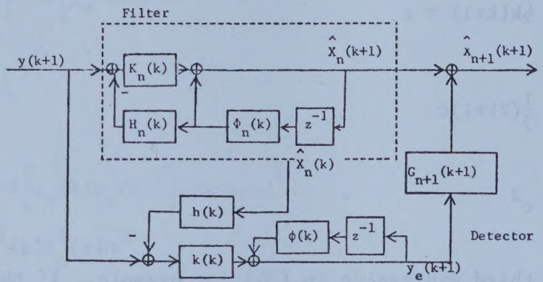

Filter
y(k+1)
$K_n(k)$
$\hat{X}_n(k+1)$
$\hat{x}_{n+1}(k+1)$
$H_n(k)$
$\Phi_n(k)$
z^{-1}
$\hat{X}_n(k)$
$G_{n+1}(k+1)$
h(k)
$\phi(k)$
z^{-1}
Detector
-
k(k)
$y_e(k+1)$

Fig. 1 Block diagram of adaptive IMF.

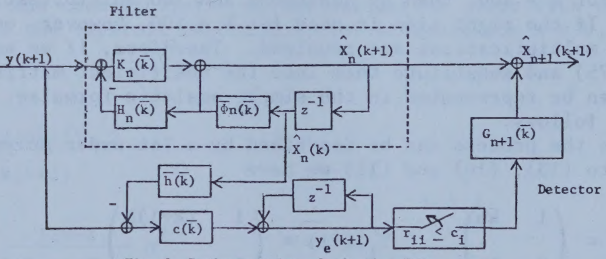

Filter
y(k+1)
$K_n(\bar{k})$
$\hat{X}_n(k+1)$
$\hat{X}_{n+1}(k+1)$
$H_n(\bar{k})$
$\bar{\Phi}n(\bar{k})$
z^{-1}
$\hat{X}_n(k)$
$G_{n+1}(\bar{k})$
$\bar{h}(\bar{k})$
z^{-1}
Detector
-
$c(\bar{k})$
$y_e(k+1)$
$r_{ii} \leq c_i$

Fig. 2 Implementation of adaptive FMF.

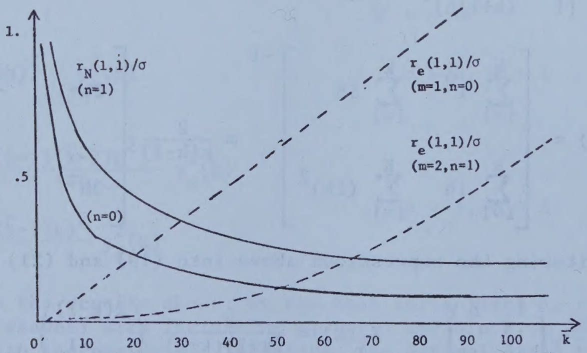


Fig. 3 Relative deviation of random error and truncate error
$(h=0.025,\ y_e(k)/\sigma=1)$.

OPTIMUM TIME VARYING FILTERS FOR TRANSIENT WAVEFORM ESTIMATION

K. B. YU, C. D. MCGILLEM
School of Electrical Engineering
Purdue University, W. Lafayette, Indiana

ABSTRACT

The use of linear time varying filters is proposed for the estimation of transient signals in the presence of an additive noise process. The minimum mean square error criterion is used and this leads to a matrix normal equation. The situation when the signal location is randomly jittering is also discussed. Performance estimates for the filters are derived. Results of testing the filters on simulated signals and noise are included and illustrate that substantial improvements in signal-to-noise ratio can be obtained with filters of this type.

INTRODUCTION

Assume that an observed waveform r(t) is the sum of a signal s(t) and a zero mean random process n(t), i.e.

$$r(t) = s(t) + n(t) \qquad 0 \leq t \leq T \tag{1}$$

The signal s(t) in (1) can be extracted from r(t) by a variety of signal processing techniques, the details of which depend on the measured or assumed characteristics of both s(t) and n(t), the form of the estimator, and the criterion to be optimized [1]. The most widely reported techniques are based on the assumptions that the noise and the signal are sample functions from stationary random process. Such an assumption and the minimum mean square error criterion lead to the Weiner filter [2].

Here we consider the case of a nonstationary signal s(t) which is a transient characterized by a distinct onset and a finite duration. The signal can be deterministic or random and is assumed to be imbedded in samples of noises n(t) which can be stationary or nonstationary. This situation occurs whenever the transient signal is generated by application of a stimulus or other excitation. In the case considered here the transient corresponds to evoked brain potentials resulting from application of sensory inputs.

OPTIMUM LINEAR FILTER

Consider a discrete time formulation in which samples of the observations are taken and are put into the following form

$$\underline{r} = \underline{s} + \underline{n} \tag{2}$$

where

$$\underline{r} = \begin{bmatrix} r(t_1) \\ \cdot \\ \cdot \\ r(t_N) \end{bmatrix} \qquad \underline{s} = \begin{bmatrix} s(t_1) \\ \cdot \\ \cdot \\ s(t_N) \end{bmatrix} \qquad \underline{n} = \begin{bmatrix} n(t_1) \\ \cdot \\ \cdot \\ n(t_N) \end{bmatrix}$$

It is desired to design a filter to process each data record to extract the signal information. In particular, an estimator of the following form is of interest.

$$\underline{\hat{S}} = \underline{A}\,\underline{r} \tag{3}$$

where $\underline{A}$ is a NxN matrix. The design procedure is to find $\underline{A}$ so as to minimize the mean square error criterion. In general, the filter will be time varying and non-causal.

Let the error of the estimator be designated as

$$\underline{e} = \underline{\hat{s}} - \underline{s} \tag{4}$$

The mean square error is then given by

$$\begin{aligned} \varepsilon &= E\{\underline{e}^T\underline{e}\} \\ &= E\{tr\underline{e}\underline{e}^T\} \\ &= E\{tr(\underline{A}\underline{r} - \underline{s})(\underline{A}\underline{r} - \underline{s})^T\} \\ &= tr\underline{A}\textstyle\sum_{rr}\underline{A}^T - 2\underline{A}\sum_{rs} + \sum_{ss} \end{aligned} \tag{5}$$

where

$$\Sigma_{rr} = E\{\underline{r}\underline{r}^T\} \qquad \Sigma_{rs} = E\{\underline{r}\underline{s}^T\} \qquad \Sigma_{ss} = E\{\underline{s}\underline{s}^T\}$$

Σ_{rr} is the autocorrelation of the data measurements, Σ_{sr} is the crosscorrelation of the data and the signal, Σ_{ss} is the autocorrelation of the signal when the signal is random and is just the outerproduct of the signal when it is nonrandom. The filter $\underline{A}$ can be derived formally and results in the following matrix formulation,

$$\frac{\partial \varepsilon}{\partial \underline{A}} = 2\underline{A}\Sigma_{rr} - 2\Sigma_{sr} = 0$$

or

$$\underline{A}\Sigma_{rr} = \Sigma_{sr} \tag{6}$$

A solution $\underline{A}$ always exists which will satisfy the above equations since there are the same number of unknowns (N^2) as the number of equations (N^2). Σ_{rr} is symmetric and positive semi-definite. If Σ_{rr} is positive definite (i.e. all the eigenvalues are positive), then the inverse of Σ_{rr} exists and $\underline{A}$ is uniquely defined by

$$\underline{A} = \Sigma_{rs}\Sigma_{rr}^{-1} \tag{7}$$

If some of the eigenvalues of Σ_{rr} are zero, then the inverse of Σ_{rr} does not exist, however it is still possible to define a generalized inverse for it [3]. (7) is still valid when Σ_{rr}^{-1} is a generalized inverse even if Σ_{rr} is a singular matrix.

The minimum mean square error can now be obtained using (6) and (7) as follows,

$$\begin{aligned} \varepsilon_{min} &= Tr\{\underline{A}\Sigma_{rr}\underline{A}^T - 2\underline{A}\Sigma_{rs} + \Sigma_{ss}\} \\ &= Tr\{\Sigma_{ss} - \Sigma_{sr}\underline{A}^T\} \qquad (8) \\ &= Tr\{\Sigma_{ss} - \Sigma_{sr}\Sigma_{rr}^{-1}\Sigma_{sr}\} \end{aligned}$$

When the signal and noise are uncorrelated, Σ_{sr} will be just Σ_{ss} and the above becomes

$$\begin{aligned}\varepsilon_{min} &= \mathrm{Tr}\{\Sigma_{ss} - \Sigma_{ss}(\Sigma_{ss} + \Sigma_{nn})^{-1}\Sigma_{ss}\} \\ &= \mathrm{Tr}\{\Sigma_{ss}(\Sigma_{ss} + \Sigma_{nn})^{-1}\Sigma_{nn}\}\end{aligned} \tag{9}$$

The theoretical performance of the filter can be evaluated by (9) using estimated values of Σ_{ss} and Σ_{nn} even before implementing the filter and testing it with experimental or simulated data.

SIGNAL WITH RANDOM JITTER

The case when the signal latency is randomly jittering can be represented by the following model

$$r(t) = s(t+\tau) + n(t) \tag{10}$$

where τ is the randomly varying time jitter. In many applications; e.g., evoked potential estimation in brain wave signal processing, it is reasonable to assume τ to be a Gaussian random variable with zero mean and standard deviation σ. In the discrete time case (10) can be expressed in the following form,

$$\underline{r} = \underline{s}' + \underline{n} \tag{11}$$

where $\underline{r}$ and $\underline{n}$ are as in (2), while $\underline{s}'$ is a random vector of $\underline{s}$ which takes the following form,

$$\underline{s}' = \begin{bmatrix} s(t_1+\tau) \\ \cdot \\ \cdot \\ s(t_n+\tau) \end{bmatrix}$$

Here we consider the desired signal to be a deterministic signal with a random jitter. The covariance matrix of the desired signal is given by,

$$\Sigma_{s's'} = E\left\{ \begin{bmatrix} s(t_1+\tau) \\ \cdot \\ \cdot \\ s(t_n+\tau) \end{bmatrix} [s(t_1+\tau) \quad \cdots \quad s(t_n+\tau)] \right\}$$

$$\simeq \left[\sum_{i=-K}^{K} s(t_i+i)s(t_j+i)P(i) \right]_{i,j}$$

$$= p_o \left[s(t_i)s(t_j) \right]_{i,j} + p_1 \left[s(t_i+1)s(t_j+1) \right]_{i,j} + p_1 \left[s(t_i-1)s(t_j-1) \right]_{i,j} + \cdots$$

$$= \qquad + p_k \left[s(t_i+k)s(t_j+k) \right]_{i,j} + p_k \left[s(t_i-k)s(t_j-k) \right]_{i,j}$$

$$= p_o \underline{s}\,\underline{s}^T + p_1 \underline{S}_L \underline{s}\,\underline{s}^T \underline{S}_R + p_1 \underline{S}_R \underline{s}\,\underline{s}^T \underline{S}_L + \cdots$$

$$+ p_k \underline{S}_L^{(k)} \underline{s}\,\underline{s}^T \underline{S}_R^{(k)} + p_k \underline{S}_R^{(k)} \underline{s}\,\underline{s}^T \underline{S}_L^{(k)} \tag{12}$$

where $\underline{S}_L$, $\underline{S}_R$ are shifting operators given by

$$\underline{S}_L = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & \bigcirc & \\ & & & & \\ & \bigcirc & & & 1 \\ & & & & 0 \end{bmatrix} \qquad \underline{S}_R = \begin{bmatrix} 0 & & & & \\ 1 & 0 & & \bigcirc & \\ & 1 & & & \\ & \bigcirc & & 0 & \\ & & & 1 & 0 \end{bmatrix}$$

and

$$p_i = \int_{(i-\frac{1}{2})\Delta T}^{(i+\frac{1}{2})\Delta T} p(\tau)d\tau \quad , \quad p(\tau) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\tau^2}{2\sigma^2}}$$

where $p(\tau)$ is the probability density of the jitter, ΔT is the sampling time interval and p_i is the probability of the waveform shifting by i units. Premultiplying and postmultiplying the $\underline{s}\ \underline{s}^T$ matrix by $\underline{S}_L$ and $\underline{S}_R$ will shift the {i,j} element of $\underline{s}\ \underline{s}^T$ to the {i-1,j-1} element. Similarly premultiplying and post multiplying the $\underline{s}\ \underline{s}^T$ matrix by $\underline{S}_R$ and $\underline{S}_L$ will shift the {i,j} element of $\underline{s}\ \underline{s}^T$ to the {i+1,j+1} element. Once the covariance matrix of the desired signal is obtained, it can be used to design the filter using (7).

EXPERIMENTAL RESULTS

In order to demonstrate the performance of the filters in a realistic manner yet retain enough control of the test parameters to permit verification of the performance, signals and noise were chosen to simulate realistic conditions. The noise was generated from random samples having a covariance matrix identical to that of experimentally measured samples of a human on-going EEG measured at scalp electrode position P_z. The signal was chosen to be the average evoked potential determined from 200 visual stimulations with a checkerboard pattern in the upper half visual field. The signal was added to samples of the noise to give a signal-to-noise ratio of -6 dB. Jitter was produced by introducing random time delays in the signal having a Gaussian distribution with zero mean and standard deviation 8 ms.

Two filters were designed to process the data records. One filter ignored the effects of signal time jitter and is called a constant latency filter while the other filter took this effect into account and is called a variable latency filter. Typical results are shown in Figure 1-6 for the filtering of signals having zero time displacement, a +12 ms time displacement and a -12 ms time displacement. To illustrate the noise waveforms of the filters the noise cancelling performance without the signal were processed as well as the signal plus noise waveforms. A sampling interval of 4 ms was employed.

Figures 1, 2, and 3 correspond to cases of signal latencies of 0, +12 and -12 ms, respectively. In each figure the left hand waveforms show the unfiltered signal plus noise (solid) and the underlying signal component (dashed). The center waveforms are the outputs of the constant latency filter (solid) and the true signal waveform (dashed). The right hand

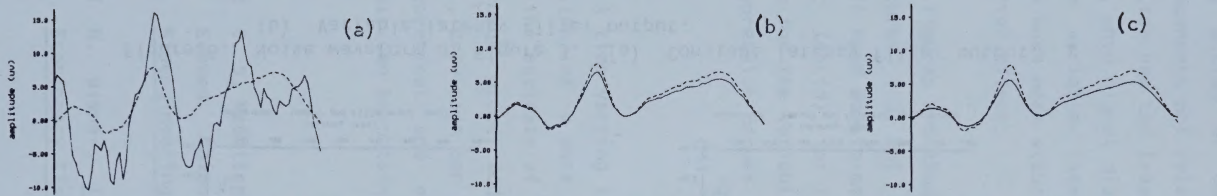


Figure 1. Signal without displacement. (a) Signal plus noise. (b) Constant latency filter output. (c) Variable latency filter output.

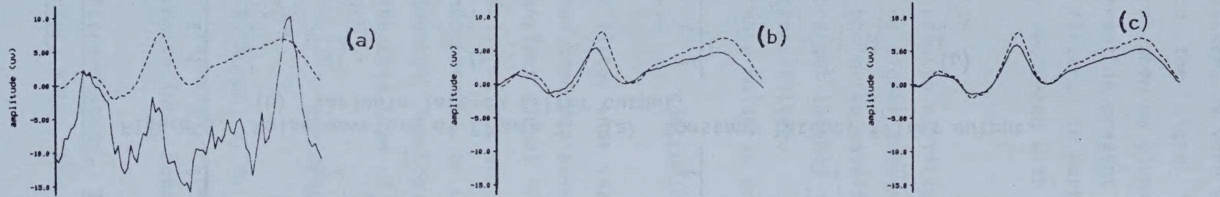


Figure 2. Signal with +12ms displacement. (a) Signal plus noise. (b) Constant latency filter output. (c) Variable latency filter output.

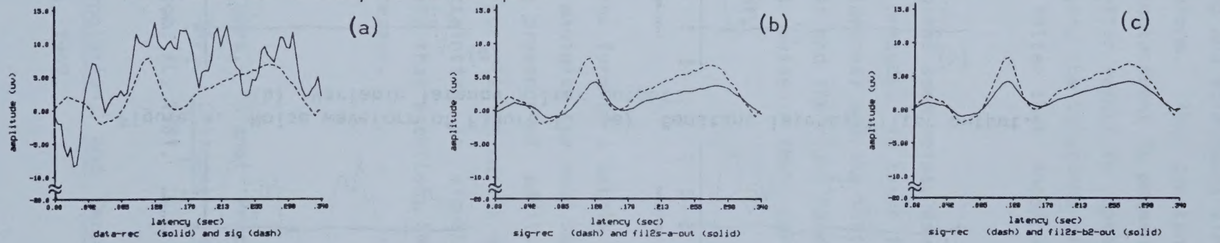


Figure 3. Signal with -12ms displacement. (a) Signal plus noise. (b) Constant latency filter output. (c) Variable latency filter output.

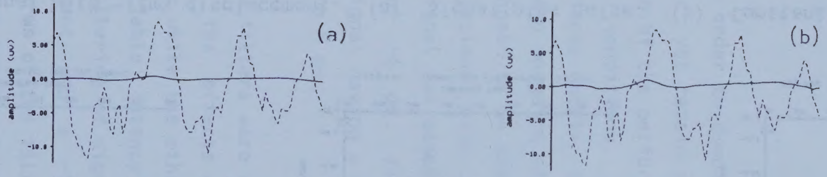


Figure 4. Noise waveform of Figure 1. (a) Constant latency filter output. (b) Variable latency filter output.

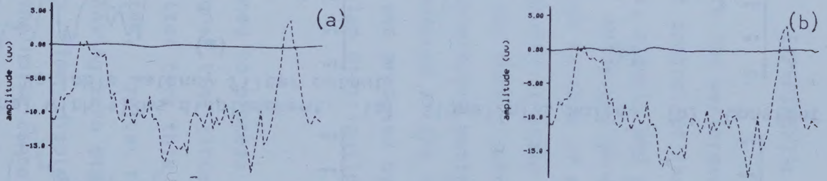


Figure 5. Noise waveform of Figure 2. (a) Constant latency filter output. (b) Variable latency filter output.

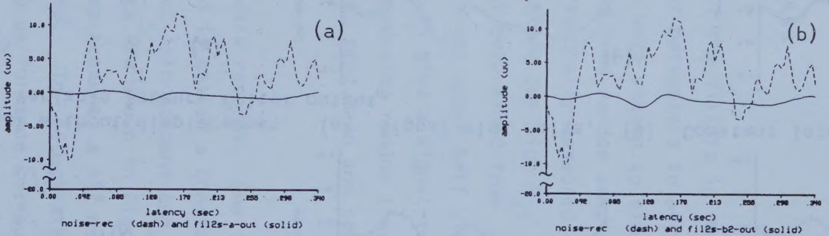


Figure 6. Noise waveform of Figure 3. (a) Constant latency filter output. (b) Variable latency filter output.

waveforms are the outputs for the variable latency filter (solid) and the true signal (dashed). It is evident from these waveforms that the filtering action of both filters is very strong and eliminates virtually all components not related to the signal waveform. The constant latency filter has the least error when no signal displacement is present. However, when signal displacement is present a better result is obtained with the variable latency filter. In particular, the locations of the peaks are much more accurately recovered with the filter that incorporates the jitter effect.

Figures 4-6 show the results of filtering the same noise waveforms as Figures 1-3 but with no signal component present. In these figures the left hand waveforms are the noise waveform (dashed) and the filtered output (solid) for the constant latency filter and the right hand waveforms are for the variable latency filter. It is evident that strong noise suppression occurs when only noise is present.

CONCLUSIONS

A time varying linear filter that takes the form of a matrix transformation of a data set has been designed to minimize the mean square error of the estimate of a transient signal in the presence of additive noise. Through proper modification of the assumed correlation matrix of the signal, provision can be incorporated for uncertainties in signal latency. Theoretical and experimental results verify that excellent SNR improvements can be obtained with this type of processor.

REFERENCES

[1] C. D. McGillem, J. I. Aunon, D. G. Childers, "Signal Processing in Evoked Potential Research, Part II," Critical Reviews in Bioengineering CRC Press Inc., Boca Raton, FL, 1981.

[2] N. Wiener, The Extrapolation, Interpolation, and Smoothing of Stationary Time Series, Wiley, New York, 1949.

[3] C. R. Rao, Linear Statistical Inference and its Applications, 1973, John Wiley & Sons, Inc.

VOLTERRA SERIES APPROACH TO FILTERING

JAMES TING-HO LO and SZE-KUI NG
Department of Mathematics
University of Maryland Baltimore County
Catonsville, Md. 21228

Abstract:

Projecting a Bayesian-type representation of the minimum variance estimate onto the Hilbert subspace generated by the n-th order Volterra series driven by the observation yields a minimum variance Volterra representation $\overset{n}{\hat{x}}$ of the same order. Its kernels are characterized by a Fredholm integral equation of the second kind, which can be viewed as "the Wiener-Hopf equation of the n-th order".

By one more projection, from $\overset{n}{\hat{x}}$ onto the subspace generated by Volterra series with separable kernels, a finite dimensional recursive filter is obtained. It is optimal for its kind. There is much flexibility in the choice of the kernels, which promises great opportunity in the design of the optimal finite-order filters.

The minimal variance estimate can also be projected directly onto the Hilbert subspace of all Fourier-Hermite (FH) series, driven by the observations, with the same index set. The projection results in a system of linear algebraic equations for the FH coefficients, the parameters of the desired approximate estimator.

The estimator consists of finitely many Wiener integrals of the observations and a memoryless nonlinear postprocessor. The postprocessor is an arithmatic combination of the Hermite polynomials evaluated at the Wiener integrals.

If a complete orthonormal set of step functions is chosen for the FH series, the approximate estimator uses only the observations at discrete times.

Acknowledgement: This work was supported in part by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant AFOSR-80-0241.

I. PROBLEM STATEMENT AND PRELIMINARIES

Consider the scalar system that is described by the following signal and sensor equations in the sense of Ito:

$$\begin{aligned} dx &= f(x,t)dt + g(x,t)dv, \\ dy &= h(x,t)dt + r_t^{1/2}dw \end{aligned} \tag{1}$$

where w and v are independent standard Wiener processes and x_0 is an independent random variable with a given distribution [1]-[3]. It is assumed that the functions f, g, and h are such that the signal equation has a unique solution and the measures μ_y and μ_w induced by y and w respectively on the measurable space $(C[0,t],\mathscr{B})$ of the continuous functions and the associated Borel field are equivalent and, almost surely with respect to μ_y,

$$\frac{d\mu_y}{d\mu_w}(y) = E_{\mu_x}(\theta^t(x,y)) > 0$$

$$\theta^t(x,y) := \exp\left[\int_0^t h(x(\tau),\tau)r_\tau^{-1}\,dy - \frac{1}{2}\int_0^t h^2(x_\tau,{}_\tau)r_\tau^{-1}d\tau\right]$$

Furthermore the conditional expectation $\hat{\phi}(x_t) := E(\phi(x_t)|y^t)$ of a measurable $\phi(x_t)$ of x_t given the measurement $y^t := \{y_\tau,\ 0 \le \tau \le t\}$ has the following representation [1]-[3]:

$$\hat{\phi}(x_t) = \sigma_t(\phi)/\sigma_t(1)$$

$$\sigma_t(\phi) := (\sigma_t(\phi))(y) := E_{\mu_x}(\phi(x_t)\theta^t(x,y))$$

where 1 is the constant function with value one everywhere. It is known [1]-[3] that $\sigma_t(\phi)$ satisfies the stochastic differential equation,

$$d\sigma_t(\phi) = \sigma_t(\mathcal{L}\phi)\,dt + r^{-1}\sigma_t(h\phi)\,dy_t,\quad \sigma_0(\phi) = E(\phi(x_0))$$

$$(\mathcal{L}\phi)(x) := f(x,t)\frac{\partial}{\partial x}\phi(x) + \frac{1}{2}g^2(x,t)\frac{\partial^2}{\partial x^2}\phi(x) \tag{2}$$

It is known that the solution of (2) satisfies the integral equation,

$$\sigma_t(\phi) = E(\phi(x_t)) + \int_0^t \sigma_s(hT_{t,s}\phi)\,dy_s$$

$$(T_{t,s}\phi)(x) := E(\phi(x_t)|x_s = x)$$

Applying the Peano-Baker procedure, it is straightforward to construct the n-th order Volterra expansion of $\sigma_t(\phi)$,

$$\sigma_t(\phi) = E[\phi(x_t)] + \int_0^t E\left[\left(hT_{t,s_1}\phi\right)(x_{s_1})\right]dy_{s_1}$$

$$+ \cdots + \int^{n-1} E\left[\left(hT_{s_{n-2},s_{n-1}}\left(hT_{s_{n-3},s_{n-2}}\left(\cdots\left(hT_{t,s_1}\phi\right)\right)\right)\right)(x_{s_{n-1}})\right]$$

$$\cdot\, d^{n-1}y$$

$$+ \int^{n} \sigma_{s_n}\left(hT_{s_{n-1},s_n}\left(hT_{s_{n-2},s_{n-1}}\left(\cdots\left(hT_{s_1,s_2}\left(hT_{t,s_1}\phi\right)\right)\cdots\right)\right)\right)d^ny$$

$$d^ky := dy_{s_k}\,dy_{s_{k-1}}\cdots dy_{s_1} \tag{3}$$

Here and in the following, we use the notation

$$\int^k (\quad)\,d^ky := \int_0^t\int_0^{s_1}\cdots\int_0^{s_{k-1}}(\quad)\,d^ky$$

Consider the set of all n-th Volterra series expansions in y^t, of which the kernels are L^2 functions, and denote it by $H_n(t)$:

$$H_n(t) := \left\{a_0(t) + \sum_{k=1}^{n}\int^k a_k(t, s_1,\ldots,s_k)\,d^ky \;\middle|\; a_k \in L^2([0, t]^k),\ k = 1,\ldots, n\right\}$$

We will call an element of $H_n(t)$, denoted by $\overset{n}{\hat{\phi}}(x_t)$, an optimal n-th order estimate of $\phi(x_t)$ given the observation y^t, if it satisfies, for every $z_n \in H_n(t)$,

$$E(\phi(x_t) - \overset{n}{\hat{\phi}}(x_t))^2 \le E(\phi(x_t) - z_n)^2$$

The uniqueness is shown.

II. PROJECTION FOR THE n-th ORDER FILTERING

Theorem 2.1: If $\sigma_t(\phi)$ has an n-th order Volterra series representation as follows:

$$(\sigma_t(\phi))(y) = \alpha_0(t) + \sum_{k=1}^{n} \int^k \alpha_k(t, s_1, \ldots, s_k) d^k y(s)$$

$$+ \int^{n+1} r_{n+1}(t, s_1, \ldots, s_{n+1}, y^{s_{n+1}}) d^{n+1} y(s) \qquad (4)$$

then the functional $\hat{\phi}(x(t) \in H_n(t)$ is an optimal n-th order estimate of x(t) if and only if

$$R_{n+1}(y) := \hat{\phi}(x_t)\sigma_t(1) - \alpha_0(t) - \sum_{k=1}^{n} \int^k \alpha_k(t, s_1, \ldots, s_k) d^k y \qquad (5)$$

is orthogonal to $H_n(t)$, with respect to μ_w, i.e. for every $z_n(y) \in H_n(t)$,

$$E(R_{n+1}(w) z_n(w)) = 0 \qquad (6)$$

III. A PRODUCT-TO-SUM FORMULA

Proposition 3.1: Let a and b be functions from $L^2([0, t]^n)$ and $L^2([0, t]^m)$ respectively. Then the product of two iterated stochastic integrals has a finite Volterra series expansion as follows:

$$\left[\int^n a(s_1, \ldots, s_n) d^n w(s)\right]\left[\int^m b(s_1, \ldots, s_m) d^m w(s)\right]$$

$$= \sum_{k=0}^{\min(n,m)} \frac{1}{k!} \int^{n+m-2k} \left[\fint^k \sum_{\pi} \bar{a}(\sigma_1, \ldots, \sigma_k, s_{\pi(1)}, \ldots, s_{\pi(n-k)})\right.$$

$$\left. \cdot \bar{b}(\sigma_1, \ldots, \sigma_k, s_{\pi(n-k+1)}, \ldots, s_{\pi(n+m-2k)}) d^k \sigma\right] d^{n+m-2k} w \qquad (7)$$

where the summation $\sum_{\pi}$ is taken over all combinations π of $n+m-2k$ elements $\{1, \ldots, n+m-2k\}$ taken $n-k$ at a time, and

$$\bar{a}(s_1, \ldots, s_n) := a(s_{\tau(1)}, \ldots, s_{\tau(n)}) \qquad (8)$$

τ := a permutation of 1,..., n such that

$$s_{\tau(1)} \geq s_{\tau(2)} \geq \cdots \geq s_{\tau(n)}.$$

$$\fint^k (\quad) d^k \sigma := \int_0^t \cdots \int_0^t (\quad) d\sigma_k \cdots d\sigma_1$$

IV. THE n-th ORDER WIENER-HOPF EQUATIONS

Assume that we have obtained the following representation of $\sigma_t(\phi)$ and $\sigma_t(1)$:

$$\sigma_t(\phi) = \alpha_0(t) + \sum_{k=1}^{n} \int^k \alpha_k(t, s_1, \ldots, s_k) d^k y + \int^{n+1} r_{n+1}(t, s_1, \ldots, s_{n+1}, y^{s_{n+1}}) d^{n+1} y \qquad (9)$$

$$\sigma_t(1) = 1 + \sum_{k=1}^{2n} \int^k \beta_k(t, s_1, \ldots, s_k) d^k y + \int_0^{2n+1} r'_{2n+1}\left(t, s_1, \ldots, s_{2n+1}, y^{s_{2n+1}}\right) d^{2n+1} y \qquad (10)$$

Let the optimal n-th order estimate $\overset{n}{\hat{\emptyset}}(x_t)$ be written as

$$\overset{n}{\hat{\emptyset}}(x_t) = b_0(t) + \sum_{k=1}^{n} \int^{k} b_k(t, s_1,\dots,s_k)d^k y \tag{11}$$

where b_k, $k = 0,\dots, n$, are unknown functions to be characterized.

By applying the product formula (7) we can expand $\overset{n}{\hat{\emptyset}}(x_t)\sigma_t(1)$ as

$$\overset{n}{\hat{\emptyset}}(x_t)\sigma_t(1) = \sum_{m=0}^{n} \sum_{i=0}^{n} \sum_{k=\max(0,i-m)}^{i} \frac{1}{k!}\int^{m}\left(\fint^{k} \sum_{\pi} \bar{b}_i \bar{\beta}_{m+2k-i} d^k\sigma\right) d^m y + 0_{n+1} \tag{12}$$

where 0_{n+1} denotes terms that are orthogonal to every $z_n \in H_n$ with respect to μ_w.

By Theorem 2.1, the kernels b_i's satisfy the equations, for $m = 0,\dots,n$ and $s_m \leq \cdots \leq s_1$

$$\sum_{i=0}^{n} \sum_{k=\max(0,i-m)}^{n} \frac{1}{k!}\left(\fint^{k} \sum_{\pi} \bar{b}_i \bar{\beta}_{m+2k-i} d^k\sigma\right) = \alpha_m \tag{13}$$

In order to set the stage for the numerical analysis in the sequel, we transform this equation into the form (16). We need summation $\sum\limits_{\pi(i,j)}$:

$$\sum_{\pi(i,j)} \bar{a}_k \bar{b}_j := \sum_{\pi} \bar{a}_k(\sigma_1,\dots,\sigma_{k-1}, s_{\pi(1)},\dots,s_{\pi(i)})\bar{b}_j(s_{\pi(i+1)},\dots,s_{\pi(i+j)})$$

and the functions $a_m :=: a_m(t,s_1,\dots,s_m)$ defined recursively by

$$\begin{aligned} a_0 &:= \alpha_0 \\ a_m &:= \alpha_m - \sum_{j=0}^{m-1} \sum_{\pi} a_j \bar{\beta}_{m-j} \end{aligned} \tag{14}$$

for $s_m \leq \cdots \leq s_1$ and $m = 1,\dots, n$. Substituting (14) into (13) yields

$$a_m = \sum_{k=0}^{n} \frac{1}{k!}\left(\fint^{k} \sum_{i=k}^{\min(k+m,n)} \sum_{\pi} \bar{b}_i \bar{\gamma}^{m}_{m+2k-i} d^k\sigma\right), \qquad m = 0,\dots, n \tag{15}$$

where $\bar{\gamma}^{m}_{m+2k-i}$ is defined recursively by

$$\bar{\gamma}^{0}_{0+2k-i} := \bar{\beta}_{2k-i}$$

$$\bar{\gamma}^{m}_{m+2k-i} := \bar{\beta}_{m+2k-i} - \sum_{j=0}^{m-1}\left(\sum_{\pi(j+k-i,m-j)} \bar{\gamma}_{j+2k-i}\bar{\beta}_{m-j}\right)\chi_{[k,k+j]}(i)$$

The last n equations of (15) are decoupled from the zeroth one and they can be written as, for $m=1,\dots,n$ and $s_m \leq \cdots \leq s_1$,

$$a_m(t,s_1,\dots,s_m) = \bar{b}_m(t,s_1,\dots,s_m) + \sum_{k=1}^{n} \frac{1}{k!}\left\{\int^{k} \sum_{i=k}^{\min(k+m,n)} \sum_{\pi} \bar{b}_i(t,\sigma_1,\dots,\sigma_k,s_{\pi(1)},\dots,s_{\pi(i-k)}) \cdot \bar{\gamma}^{m}_{m+2k-i}(t,\sigma_1,\dots,\sigma_k,s_{\pi(i-k+1)},\dots,s_{\pi(m)})\,d^k\sigma\right\} \tag{16}$$

This equation can be viewed as "the n-th order Wiener-Hopf equation". We note that each term on the right side of (16) is a finite sum of integral operators with square-integrable kernels (under the assumption that $\beta_k \in L^2([0,t]^k)$. Hence it is a compact linear (or completely continuous) operator [4]-[5] and equation (16) is a Fredholm integral equation of the second kind. This is a kind of operator equation, which has been extensively studied in functional analysis. There exists quite a few numerical methods [7]-[8], [13]-[14] to solve it.

V. OPTIMAL FINITE-ORDER FILTERS

In the following, we will take a different approach. Rather than trying to realize $\overset{n}{\emptyset}_j$, we will realize the optimal estimate $\hat{\hat{\emptyset}}$ that can be expressed in the form

$$\hat{\hat{\emptyset}}(x_t) = b_0(t) + \int_0^t b_1(t,s)dy(s) + \sum_{k=2}^{n}\int^{k} \sum_{(i_1,\dots,i_k)\in I_k} c(t,k;i_1,\dots,i_k) \cdot \phi_{i_1}(s_1)\cdots\phi_{i_k}(s_k)d^k y \tag{17}$$

where c's are arbitrary constants (t is fixed here); $\{\phi_k, k = 0, 1,\dots\}$ is a complete sequence of independent functions in $L^2([0,t])$; and the summation $\sum$ is taken over a given finite set I_k of k-tuples of nonnegative integers. We emphasize that I_k is given and fixed for k = 2,..., n. The reason why we choose the functions of the form $\sum_{I_k} c\phi_{i_1}\cdots\phi_{i_k}$, as the kernels is two-fold. First, the set of all such functions is dense in $L^2([0,t]^k)$. Secondly, the iterated stochastic integral, $\int^k \sum_{I_k} c\phi_{i_1}\cdots\phi_{i_k}d^k y$, can be easily realized with bilinear differential equations.

We note that $\hat{\hat{\emptyset}} \in H_n$ and hence $E(\emptyset - \hat{\hat{\emptyset}})^2 = E(\emptyset - \overset{n}{\hat{\emptyset}})^2 + E(\overset{n}{\hat{\emptyset}} - \hat{\hat{\emptyset}})^2$. Therefore the optimal estimate that can be expressed in the form (17) can be obtained by minimizing $E(\overset{n}{\hat{\emptyset}} - \hat{\hat{\emptyset}})^2$ over all the constants, c's. The criterion $E(\overset{n}{\hat{\emptyset}} - \hat{\hat{\emptyset}})^2$ can be straightforwardly calculated as follows.

$$E(\overset{n}{\hat{\emptyset}} - \hat{\hat{\emptyset}})^2 = E\left(\sum_{k=2}^{n}\int^{k}(\bar{b}_k - \sum_{I_k} c\phi_{i_1}\cdots\phi_{i_k})d^k y\right)^2$$

$$f_k := \bar{b}_k - \sum_{I_k} c\phi_{i_1}\cdots\phi_{i_k} =: \bar{b}_k - \sum_{I_k} c\Phi(k)$$

$$F_m := \sum_{k=0}^{n}\frac{1}{k!} -\!\!\!\!\!\!\int^{k} \sum_{i=\max(k,m-n+2k,2)}^{\min(m-2+2k,m+k,n)} \sum_{\pi} \bar{f}_i\bar{f}_{m+2k-i}d^k\sigma$$

$$\left(\sum_{k=2}^{n}\int^{k} f_k d^k y\right)^2 = \sum_{m=0}^{2n}\int^{m} F_m d^m y$$

$$E(\overset{n}{\hat{\emptyset}} - \hat{\hat{\emptyset}})^2 = \sum_{m=0}^{2n}\int_0^t\int_0^{s_1}\cdots\int_0^{s_{m-1}} F_m\beta_m d^m s$$

$$-\!\!\!\!\!\!\int^{k}\sum_{\pi}\bar{f}_i\bar{f}_{m+2k-i}d^k\sigma = -\!\!\!\!\!\!\int^{k}\Big[\sum_{\pi}\bar{b}_i\bar{b}_{m+2k-i} - \sum_{I_i} c_i \sum_{\pi}\bar{\Phi}(i)\bar{b}_{m+2k-1}$$

$$- \sum_{I_{m+2k-1}} c_{m+2k-1}\sum_{\pi}\bar{b}_i\bar{\Phi}(m + 2k - 1)$$

$$+ \sum_{I_i}\sum_{I_{m+2k-1}} c_i c_{m+2k-1}\sum_{\pi}\bar{\Phi}(i)\bar{\Phi}(m+2k-1)\Big]d^k\sigma$$

Defining the operator Q and $\sum_{G}$ by

$$Q_{mki}(\alpha,\beta) := \int_0^t\int_0^{s_1}\cdots\int_0^{s_{m-1}}\beta_m\left(\int^{k}\sum_{\pi}\bar{\alpha}_i\beta_{m+2k-1}d^k\sigma\right)d^m s \tag{18}$$

$$\sum_{G} := \sum_{m=0}^{2n}\sum_{k=0}^{n}\frac{1}{k!}\sum_{i=\max(k,m-n+2k,2)}^{\min(m-2+2k,m+k,n)} \tag{19}$$

it is easy to see that the criterion can be expressed as a quadratic function in c's:

$$E(\overset{n}{\hat{\emptyset}} - \hat{\hat{\emptyset}})^2 = \sum_{G} Q_{mki}(\bar{b},\bar{b}) - \sum_{G}\sum_{I_i} c_i Q_{mki}(\Phi,\bar{b}) - \sum_{G}\sum_{I_{m+2k-1}} c_{m+2k-1}Q_{mki}(\bar{b},\Phi)$$

$$+ \sum_{G}\sum_{I_i}\sum_{I_{m+2k-1}} c_i c_{m+2k-1}Q_{mki}(\Phi,\Phi) \tag{20}$$

The problem of minimizing such a quadratic function with or without constraints on c's is usually simple. The resulting estimate $\hat{\hat{\emptyset}}$ given in (20) is optimal at each time instant t with respect to the given sets I_k, $k = 2,\ldots,n$, which can be arbitrarily chosen beforehand. We note that the total number of c's in the above equation is equal to $\sum_{k=2}^{n}$ (the number of k-tuples in I_k).

We choose the form (17), mainly because it is realizable by finite-dimensional differential equations.

VI. AN ALTERNATIVE APPROACH — FOURIER-HERMITE SERIES

Let $\{\phi_i, i = 1,2,\dots\}$ be a complete orthonormal set (CONS) on $[0,t]$ and consider the multiple Wiener integral,

$$F(t,k,\xi,w) := G_k(t,\phi_{\xi_1}\cdots\phi_{\xi_k},w) \tag{21}$$

where ξ_i is a nonnegative integer for $i = 1,\dots k$ and $\xi_1 \leq \cdots \leq \xi_k$. The set of all such integrals, which are orthonormal, was called the Fourier-Hermite (FH) set by Cameron and Martin [8]. It was shown in [8] that if $A(w^t)$ is any functional for which $E[A^2(w^t)] < \infty$, then it can be approximated arbitrarily close by a FH series in the form

$$A(w^t) = \sum_{k=0}^{n} \sum_{\xi\varepsilon\underset{\sim}{\xi}_k} a(k;\xi)F(t,k,\xi,w) \tag{22}$$

$$a(k;\xi) := E[A(w^t)F(t,k,\xi,w)]$$

where $\underset{\sim}{\xi}_k$ is a finite subset of $\{\xi = [\xi_1,\dots,\xi_k] \mid \xi_1 \leq \cdots \leq \xi_k,\ \xi_i$ is a nonnegative integer for $i = 1,\dots, k\}$.

Given index sets $\underset{\sim}{\xi} = \underset{\sim}{\xi}_1 \cup\cdots\cup\underset{\sim}{\xi}_n$ and $\underset{\sim}{\eta}_1\cup\cdots\cup\underset{\sim}{\eta}_{2n}$, the finite FH series representations of $\sigma_t(\emptyset)$ and $\sigma_t(1)$ can be written as

$$\sigma_t(\emptyset) = \alpha(t,0) + \sum_{k=1}^{n}\sum_{\xi\varepsilon\underset{\sim}{\xi}_k} \alpha(t,k,\xi)F(t,k,\xi,y) + R(y^t,\underset{\sim}{\xi}) \tag{23}$$

$$\sigma_t(1) = \beta(t,0) + \sum_{k=1}^{2n}\sum_{\eta\varepsilon\underset{\sim}{\eta}_k} \beta(t,k,\eta)F(t,k,\eta,y) + R(y^t,\underset{\sim}{\eta}) \tag{24}$$

where the coefficients can be calculated by

$$\alpha(t,k;\xi) = \frac{1}{k!}\int^{k}\bar{\alpha}_k(t,s_1,\dots,s_k)\sum_{\tau}\phi_{\tau(\xi_1)}(s_1)\cdots\phi_{\tau(\xi_k)}(s_k)d^ks$$

$$= \int^{k}\bar{\alpha}_k(t,s_1,\dots,s_k)\prod_{j=1}^{k}\phi_{\xi_j}(s_j)d^ks$$

$$\beta(t,k;\eta) = \int^{k}\bar{\beta}_k(t,s_1,\dots,s_k)\prod_{j=1}^{k}\phi_{\eta_j}(s_j)d^ks$$

where $\sum_{\tau}$ is over all permutations τ of $\xi_1,\dots, \xi_k$.

$R(y^t,\underset{\sim}{\xi})$ is the remainder term orthogonal to $F(t,k,\xi,y)$ w.r.t. μ_w $\forall\xi \in \underset{\sim}{\xi}_k$, $k = 1,\dots, n$, i.e.

$$E(R(\underset{\sim}{w}^t,\underset{\sim}{\xi})F(t,k,\xi,w)) = 0, \qquad \forall \xi \in \underset{\sim}{\xi}_k,\ k = 1,\ldots, n.$$

Note that $E(R(\underset{\sim}{y}^t,\underset{\sim}{\xi})F(t,k,\xi,y)) \neq 0$.

Consider the set of all the FH series indexed by $\underset{\sim}{\xi}$ and denote it by $H_n(t,\underset{\sim}{\xi})$:

$$H_n(t,\underset{\sim}{\xi}) := \left\{ \sum_{k=0}^{n} \sum_{\xi\varepsilon\underset{\sim}{\xi}_k} a(t,k;\xi)F(t,k,\xi,y) \,\middle|\, a(t,k;\xi) \text{ are real numbers} \right\}$$

We will call an element of $H_n(t,\underset{\sim}{\xi})$ the $\underset{\sim}{\xi}$-optimal estimate of $\emptyset(x)$ and denote it by $\underset{\sim}{\xi}(\emptyset) := \underset{\sim}{\xi}_t(\emptyset(x))$, if

$$E(\underset{\sim}{\xi}_t(\emptyset(x)) - \emptyset(x))^2 \leq E(z - \emptyset(x))^2, \quad \forall z \in H_n(t,\xi)$$

Theorem 6.1: The functional $z_n(y) \in H_n(t,\underset{\sim}{\xi})$ is the $\underset{\sim}{\xi}$-optimal estimate of $\emptyset(x)$ if and only if

$$D(y) := z_n(y)\sigma_t(1) - \sum_{k=0}^{n} \sum_{\xi\varepsilon\underset{\sim}{\xi}_k} \alpha(t,k;\xi)F(t,k,\xi,y) \tag{25}$$

is orthogonal to $H_n(t,\underset{\sim}{\xi})$, w.r.t. μ_w, i.e. for every $z(y) \in H_n(t,\underset{\sim}{\xi})$,

$$E(D(w)z(w)) = 0 \tag{26}$$

Given an index set $\underset{\sim}{\xi}$, let the $\underset{\sim}{\xi}$-optimal estimate $\underset{\sim}{\xi}_t(\emptyset(x))$ be written as

$$\underset{\sim}{\xi}_t(\emptyset) = b(t,0) + \sum_{k=1}^{n} \sum_{\xi\varepsilon\underset{\sim}{\xi}_k} b(t,k;\xi)F(t,k,\xi,y) \tag{27}$$

where $b(t,k,\xi)$ for $\xi\varepsilon\underset{\sim}{\xi}_k$ and $k = 1,\ldots, n$, are unknown functions of t to be determined.

We will apply Theorem 6.1 to project $\hat{\emptyset}(x) = \sigma_t(\emptyset)/\sigma_t(1)$ onto $H_n(t,\underset{\sim}{\xi})$. The projection can be obtained by considering all the FH components of $\sigma_t(1)$ that affect each such ζ-component of the product $\underset{\sim}{\xi}_t(\emptyset)\sigma_t(1)$ that $\zeta\varepsilon\underset{\sim}{\xi}$. The index set for all those FH components of $\sigma_t(1)$ will be denoted $\underset{\sim}{\eta}$.

There is a bijective relationship between index vectors and products of ϕ's, namely $[i_1,\ldots,i_k] \leftrightarrow \phi_{i_1}\cdots\phi_{i_k}$, where ϕ's are regarded as independent variables and $i_1 \leq\cdots\leq i_k$. Let $G := \mathrm{GCD}(\phi_{\xi_1}\cdots\phi_{\xi_{k_1}}, \phi_{\zeta_1}\cdots\phi_{\zeta_{k_3}})$, $F := \phi_{\xi_1}\cdots\phi_{\xi_{k_1}}/G$, and $H := \phi_{\zeta_1}\cdots\phi_{\zeta_{k_3}}/G$. If the G power is ℓ, then there are less than or equal to 2^ℓ different factors (including 1) of it. Taking the square of each factor and multiplying it to the $F{\cdot}H$ yields the set $\underset{\sim}{\eta}(\xi,\zeta)$ of all the η that make $P := F(t,k(\xi),\xi,y)F(t,k(\eta),\eta,y)$ have a ζ-component.

When $\eta\varepsilon\underset{\sim}{\eta}(\xi,\zeta)$, we see that

$$\text{the } \zeta\text{-component of } P = C(\xi,\eta,\zeta)F(t,k(\zeta),\zeta,y)/[(k(\xi) + k(\xi) - k(\zeta))/2]! \tag{28}$$

where $k(\xi)$, $k(\eta)$, and $k(\zeta)$ are the dimension of ξ,η, and ζ, respectively.

Denote by γ the index vector associated with $(\phi_{\xi_1}\cdots\phi_{\xi_{k(\xi)}}\cdot \phi_{\eta_1}\cdots\phi_{\eta_{k(\eta)}}/\phi_{\zeta_1}\cdots\phi_{\zeta_{k(\zeta)}})^{1/2}$. A formula for $C(\zeta,\eta,\zeta)$ is the following:

$C(\xi,\eta,\zeta)$ = (# ways that the components of γ can be drawn from those of ξ)

$\cdot$(# ways that the components of γ can be drawn from those of η)

$\cdot$((# components of γ)!)

$\cdot$((# components of γ)!/(# permutations of the components of γ with subscripts removed))

Lemma 6.1: A product-to-sum formula for the finite FH series is

$$\left[\sum_{k=0}^{n}\sum_{\xi\varepsilon\underset{\sim}{\xi}_k}\gamma_1(t,k;\xi)F(t,k,\xi,y)\right]\left[\sum_{k=0}^{n}\sum_{\eta\varepsilon\underset{\sim}{\eta}_k}\gamma_2(t,k;\eta)F(t,k,\eta,y)\right]$$

$$=\sum_{\zeta\varepsilon\underset{\sim}{\zeta}(\underset{\sim}{\xi},\underset{\sim}{\eta})}\left[\sum_{k=0}^{n}\sum_{\xi\varepsilon\underset{\sim}{\xi}}\ \sum_{\eta\varepsilon\underset{\sim}{\eta}\cap\underset{\sim}{\eta}(\xi,\zeta)}\frac{C(\xi,\eta,\zeta)\gamma_1(t,k;\xi)\gamma_2(t,k(\eta);\eta)}{[(k+k(\eta)-k(\zeta))/2]!}\right]F(t,k(\zeta),\zeta,y)$$

$$\underset{\sim}{\zeta}(\underset{\sim}{\xi},\underset{\sim}{\eta}):=\bigcup_{\xi\varepsilon\underset{\sim}{\xi}}\ \bigcup_{\eta\varepsilon\underset{\sim}{\eta}}\underset{\sim}{\zeta}(\xi,\eta)$$

Applying Lemma 6.1 and setting ζ-components of $\hat{\emptyset}(x_t,\underset{\sim}{\xi})\sigma_t(1)$ and $\sigma_t(\emptyset)$ equal, we obtain

$$\left[\sum_{k=0}^{n}\sum_{\xi\varepsilon\underset{\sim}{\xi}_k}\ \sum_{\eta\varepsilon\underset{\sim}{\eta}(\xi,\zeta)}\frac{C(\xi,\eta,\zeta)b(t,k,\xi)\beta(t,k(\eta),\eta)}{((k+k(\eta)-k(\zeta))/2)!}\right]F(t,k(\zeta),\zeta,y)$$

$$=\alpha(t,k(\zeta),\zeta)F(t,k(\zeta),\zeta,y)$$

Hence for every $\zeta\varepsilon\underset{\sim}{\xi}_k$, $k=0,\dots,\eta$,

$$\sum_{k=0}^{n}\sum_{\xi\varepsilon\underset{\sim}{\xi}_k}\ \sum_{\eta\varepsilon\underset{\sim}{\eta}(\xi,\zeta)}\frac{C(\xi,\eta,\zeta)\beta(t,k(\eta),\eta)}{[(k+k(\eta)-k(\zeta))/2]!}\,b(t,k,\xi)=\alpha(t,k(\zeta),\zeta)$$

We note that these form an algebraic system of simultaneous linear equations, in which the number of unknowns $b(t,k,\xi)$ is the same as that of the equations.

Remark: $F(t,k,\xi,y)$ can be written as

$$F(t,k,\xi,y)=G_k\left(t,\ \prod_{j=1}^{k}\phi_{\xi_j},y\right)=\prod_{i=1}^{q}H_{k_i}\left(\int_0^t\phi_{\lambda_i}\,dy\right)$$

where H_{k_i} are the Hermite polynomials of order k_i.

Thus we have reduced the $\underset{\sim}{\xi}$-optimal estimation (27) to the calculation of the Wiener integrals, $\int_0^t\phi_{\lambda_i}\,dy$. The $\underset{\sim}{\xi}$-optimal estimate $\underset{\sim}{\xi}_t(\emptyset)$ can be obtained by processing these Wiener integrals through a memoryless nonlinear system consisting essentially of finite many Hermite polynomials H_{k_i}.

REFERENCES

[1] R. S. Bucy and P. D. Joseph, Filtering for Stochastic Processes with Applications to Guidance, Interscience, N. Y., 1968.

[2] G. Kallianpur, Stochastic Filtering Theory, Springer-Verlag, N. Y., 1980.

[3] R. S. Liptser and A. N. Shiryayev, Statistics of Random Processes: I General Theory, Springer-Verlag, N. Y., 1977.

[4] L. Liusternik and V. Sobolev, Elements of Functional Analysis, Ungar Publishing Co., 1965.

[5] K. Yosida, Functional Analysis, Springer Verlag, N. Y., 1974.

[6] M. A. Krasnoselskii et al., Approximation Solution of Operator Equations, Wolters-Noordhoff Publishing, Groningen, 1972.

[7] S. G. Mikhlin, Variational Methods in Mathematical Physics, Pergamon Press, London, 1964.

[8] R. H. Cameron and W. T. Martin, "The Orthogonal Development of Non-linear Functionals in Series of Fourier-Hermite Functionals", Ann. of Math. 48, 1947, 385-389.

[9] R. W. Brockett, "Volterra Series and Geometric Control Theory", Automatica, Vol. 12, 167-176.

[10] N. Wiener, Nonlinear Problems in Random Theory, The M.I.T. Press, Massachusetts, 1958.

[11] M. Schetzen, The Volterra and Wiener Theories of Nonlinear Systems, John Wiley and Sons, Inc., 1980.

[12] K. Ito, "Multiple Wiener Integrals", J. Math. Soc. Japan 3, 157-169.

[13] R. S. Bucy, "Nonlinear Filtering", I.E.E.E. Trans. Automatic Control (1965), p. 198.

[14] R. E. Kalman and R. S. Bucy, "New Results in Linear Filtering and Prediction Theory", J. Basic Eng. ASME, Ser. D., 83 (1961), 95-108.

CONVERGENCE OF STOCHASTIC APPROXIMATION ALGORITHMS WITH LINEAR FILTERS

HAI HUANG*
Department of Systems Science and Mathematics
Washington University
St. Louis, MO 63130

ABSTRACT

We consider the general form of the stochastic approximation algorithm with a linear filter: $X_{n+1} = X_n + a_n\psi_n$ and $\psi_n = \alpha\psi_{n-1} + \beta G(X_n,\xi_n)$, where G is not necessarily additive in ξ_n, $a_n = 1/(n+1)^{\varepsilon}$, or $a_n \equiv a > 0$. If $g(x) = EG(x,\xi_n)$ and θ is an asymptotically stable equilibrium point of $\dot{x} = g(x)$, then the asymptotic properties of $\{X_n\}$ and $\{X_n-\theta)/\sqrt{a_n}\}$ are developed. In particular, it is shown that X_n converges to θ with probability one, when $a_n = 1/(n+1)^{\varepsilon}$, and the tail part of $\{(X_n-\theta)/\sqrt{a_n}\}$ can be approximated by a linear stationary Gaussian diffusion, when the step size a_n is very small.

1. INTRODUCTION

This paper is concerned with asymptotic properties of stochasti approximation algorithms of the type

$$\begin{cases} X_{n+1} = X_n + a_n\psi_n, & X_n \in \mathbb{R}^r, \\ \psi_n = \alpha\psi_{n-1} + \beta G(X_n,\xi_n), & X_0 \text{ and } \psi_{-1} \text{ are given,} \end{cases} \tag{1.1}$$

where $a_n = 1/(n+1)^{\varepsilon}$, $\frac{1}{2} < \varepsilon < 1$, or $a_n \equiv a > 0$, for all n, α and β are constants, $|\alpha|<1$, $\beta > 0$, and $\{\xi_n\}$ is a stationary sequence. When $\alpha = 0$ and $\beta = 1$, the algorithm (1.1) reduces to the Robbins-Monro type algorithm which corresponds to the steepest descent method of nonlinear programming and was studied in [1], [2], [3], and [5]. Algorithms of the type (1.1) correspond to the conjugate gradient method and arise in applications to both identification theory and adaptive system theory [4] and [5]. Such correspondence and applications are the motivation for this work.

Let $g(x) = EG(x,\xi_n)$ and $h(x,\xi_n) = G(x,\xi_n) - g(x)$. Suppose θ is an asymptotically stable equilibrium point of $\dot{x} = g(x)$. Suppose $\{X_n\}$ is the sequence generated by (1.1). The convergence of $\{X_n\}$ to θ and the asymptotic behavior of the tail sequence $\{(X_{n+N} - \theta)/\sqrt{a_{n+N}}, n \geq 0\}$ are treated in this paper.

2. PROBLEM FORMULATION AND ASSUMPTIONS

In this paper, K denotes a constant whose value may change from usage

---

*This work was supported by the National Science Foundation under Grant NSF-ECS 8105978. September 21, 1981

to usage. Without loss of generality, *let $\theta = 0$*. Let $S = [c_1,d_1]\times\dots\times[c_r,d_r]$, where $-\infty < c_i < 0 < d_i < \infty$, for all $i=1,\dots,r$.

Instead of assuming that 0 is the globally asymptotically stable equilibrium point of $\dot{x} = g(x)$ and the sequence $\{X_n\}$ generated by (1.1) is uniformly bounded (see [1] and [5]), we consider the following algorithm

$$\begin{cases} X_{n+1} = X_n + a_n\psi_n + \tau_n \quad , & X_n \in \mathbb{R}^r, \\ \psi_n = \alpha\psi_{n-1} + \beta g(X_n) + \beta h(X_n,\xi_n) \quad , & X_0 \text{ and } \psi_{-1} \text{ are given,} \end{cases} \tag{2.2}$$

where τ_n is the projection vector (see [1]) such that $X_n + a_n\psi_n + \tau_n$ is the closest point on S to $X_n + a_n\psi_n$. Thus the sequence $\{X_n\}$ generated by (2.1) is confined in the bounded region S. When $a_n = 1/(n+1)^\varepsilon$, $\frac{1}{2} < \varepsilon < 1$, in Theorem 1 it will be shown that X_n converges to 0 with probability one and $E|X_n|^2 \le a_n K$. Next, following the general approach of [1], [2], and [3], a sequence of stochastic processes $\{U^N(\cdot)\}$ is defined as follows. Let $t_n = \sum_{i=0}^{n-1} a_i$, $t_0 = 0$, and $U_n = X_n/\sqrt{a_n}$. Set $U^N(0) = U_N$ and $U^N(t) = U_{N+n}$ on $[t_{N+n} - t_N, t_{N+n-1} - t_N)$. Thus $U^N(\cdot)$ is a process whose sample paths are piecewise constant and in $D^r[0,\infty)$, the space of $\mathbb{R}^r$ - valued functions which are right continuous on $[0,\infty)$ and have left-hand limits on $(0,\infty)$. In Theorem 2 it will be shown that $\{U^N(\cdot)\}$ converges weakly to the stationary Gaussian diffusion (3.14), as N goes to $+\infty$. It yields the convergence in distribution of $\{U_n = X_n/\sqrt{a_n}, n \ge 0\}$ to a Gaussian distribution (the stationary initial distribution of (3.14)) whose covariance is proportional to $\beta/(1-\alpha)$. When $a_n \equiv a > 0$ and a is very small, similar results will also be obtained.

Remark on weak convergence. Billingsly [6] is an excellent reference. The space $D^r[0,T]$, $0 < T < \infty$, is endowed with the usual Skorohod topology, with which it is a complete separable metric space [6,§14]. Convergence in $D^r[0,\infty)$ occurs if, for some sequence $\{T_\ell\}$ goes to $+\infty$, it occurs (for the truncated functions) in each $D^r[0,T_\ell]$.

Assumptions

A1. $\{\xi_n, n \ge 0\}$ is stationary, bounded by some constant K, uniformly in n, and ϕ-miximg (see [6]) with $\sum_{j=0}^{\infty} \phi(j) < \infty$.

A2. $g(x)$, $g_x(x)$, $g_{xx}(x)$, $h(x,\xi)$, $h_x(x,\xi)$, and $h_{xx}(x,\xi)$ are measurable with respect to (x,ξ) and are continuous with respect to x for each ξ. (Note that g_x and h_x denote the derivatives of g and h with respect to x.)

A3. $g(0) = 0$. All of the eigenvalues of $g_x(0)$ have negative real parts. $x'g(x) \le -\gamma x'x$, for all x in S, where x' denotes the transpose of x and γ is a positive constant.

A4. $h(0,\xi_n) = \sum_{\ell=0}^{\infty} \lambda^{\ell} \eta_{n-\ell}$, where λ is a constant, $|\lambda|<1$, and $\{\eta_n, -\infty<n<\infty\}$ is a sequence of independent and identically distributed random variables with $E\eta_n \equiv 0$ and $E|\eta_n|^2 < \infty$.

3. CONVERGENCE RESULTS

Let E_n denote the expectation conditional on $\{\xi_j, j \le n\}$. Set

$$\psi_n(x) = \sum_{j=0}^{\infty} \alpha^j \beta(g(x) + h(x,\xi_{n-j})) \text{ , for all } n \ge 0, \text{ and} \tag{3.1}$$

$$\bar{\psi}_{n,j}(x) = \alpha^j \psi_n + \sum_{i=0}^{j-1} \alpha^i \beta(g(x) + h(x,\xi_{n+j-i})) \text{ , for all } j \ge 0, \tag{3.2}$$

where x is in S. Note that $\bar{\psi}_{n,0}(x) \equiv \psi_n$.

It is easy to see that ψ_n, $\psi_n(x)$, and $\bar{\psi}_{n,j}(x)$ are bounded by some constant K, uniformly in n, j, and x in S.

Theorem 1. Assume (A1), (A2), and (A3). Let $a_n = 1/(n+1)^{\varepsilon}$, $\frac{1}{2} < \varepsilon < 1$. Then $E|X_n|^2 \le a_n K$, for all n, and X_n converges to 0 with probability one, where K is a constant.

Proof. By (A1), (3.3) is well-defined.

$$v(x,n) = \sum_{j=0}^{\infty} a_{n+j} \, E_n[\bar{\psi}_{n,j}(x) - E\psi_{n+j}(x)]. \tag{3.3}$$

Moreover,

$$|v(x,n)| \le a_n K \text{ , for all } n, \text{ for all } x \in S. \tag{3.4}$$

Define a new sequence $\{Y_n, n \ge 0\}$ by

$$Y_n = X_n + v(X_n,n). \tag{3.5}$$

It is easy to see that $\{Y_n\}$ satisfies the following algorithm:

$$Y_{n+1} = Y_n + a_n \frac{\beta}{1-\alpha} g(Y_n) + a_n m_n + a_n^2 \varepsilon_n + \tau_n , \tag{3.6}$$

where m_n and ε_n are bounded by a constant K, uniformly in n, and $\{m_n, n \ge 0\}$ is a martingale difference sequence with respect to E_n, namely, $E_n m_n = 0$ and $E_{n+1} m_n = m_n$.

Since $X_n' \tau_n \le 0$ if a_n is very small, there is an integer N such that

$|Y_n|^2$ satisfies the following recursive inequality:

$$E_n|Y_{n+1}|^2 \leq (1-a_nK_1)|Y_n|^2 + a_n^2K_2, \text{ whenever } n \geq N. \tag{3.7}$$

Interating (3.7) and taking expectations of the both sides yields

$$E|Y_{N+n}|^2 \leq \prod_{j=N}^{n+n-1} (1-a_jK_1)\ E|Y_N|^2 + \sum_{j=N}^{N+n-1} \prod_{i=j+1}^{N+n-1} (1-a_jK_1)a_j^2K_2.$$

It can be shown that the right-hand side of (3.8) is bounded above by $a_{N+n}K$, for some K. Hence

$$E|X_n|^2 \leq a_nK \text{ , for all n.} \tag{3.9}$$

(3.7) also yields

$$E_n|Y_{n+1}|^2 \leq |Y_n|^2 + a_n^2K_2 \text{ , whenever } n \geq N. \tag{3.10}$$

Set

$$q_n = |Y_n|^2 + \sum_{j=n}^{\infty} a_j^2K_2 \quad , \quad \text{for all } n \geq N. \tag{3.11}$$

Then $\{q_n, n \geq N\}$ is a nonnegative supermartingale satisfying $Eq_N < \infty$. Because of the Martingale Convergence Theorem, it implies that q_n converges with probability one. Hence Y_n and X_n converge with probability one. (3.9) implies that X_n converges to 0 with probability one. Q.E.D.

Similarly, the following result holds.

Theorem 1'. Assume (A1), (A2), and (A3). Let $a_n \equiv a > 0$. Then $E|X_n^a|^2 \leq aK$, for all $n \geq N_a$, for all $a \varepsilon (0,a_0)$, where $\{X_n^a\}$ denotes the sequence $\{X_n\}$ while $a_n \equiv a$, N_a is any integer such that $\exp(-(a\gamma/2)N_a) \leq aK$, a_0 is a constant, and K is a constant independent of a.

Let $U_n = X_n/\sqrt{a_n}$ and $Z_n = Y_n/\sqrt{a_n}$. From (3.6) and (A4), Z_n satisfies the following algorithm

$$Z_{n+1} = Z_n + a_n\left(\frac{\beta}{1-\alpha} g_x(0) + p_n\right)Z_n + \sqrt{a_n}\zeta_n + o(a_n)\theta_n \tag{3.12}$$

where

$$\zeta_n = \sqrt{\frac{a_n}{a_{n+1}}}\left[\sum_{j=1}^{\infty}\sum_{i=0}^{j-1} \frac{a_{n+j}}{a_n} \alpha^i \beta\lambda^{j-i-1}\right]\eta_{n+1} \ ,$$

$\{p_n\}$ is a martingale difference sequence with respect to E_n, $|p_n| \leq K$, and $E|\theta_n| \leq K$, uniformly in n. Note that the projection vector τ_n does not appear in (3.12), since $\{X_n\}$ and $\{Y_n\}$ will stay in a small neighborhood of 0, when n is very large and the step size is very small. Moreover,

$$|Z_n - U_n| \leq \sqrt{a_n} K. \tag{3.13}$$

Let $U^N(t) = U_{N+n}$ on $[t_{N+n} - t_N, t_{N+n+1} - t_{N+n})$, where $t_n = \sum_{j=0}^{n-1} a_j$, $t_0 = 0$, and $a_j = 1/(1+j)^\varepsilon$, $\frac{1}{2} < \varepsilon < 1$. Let $U_n^a = U_n$ and $U^a(t) = U^a_{N_a+n}$ on $[na, na + a)$, when $a_n \equiv a$. Note that N_a is an integer satisfying the condition of the Theorem 1'.

Theorem 2. Assume (A1) ~ (A4). $U^N(\cdot)$ and $U^a(\cdot)$ converge weakly to the stationary solution of the following Itô type equation

$$dU = \frac{\beta}{1-\alpha} g_x(0) U dt + \frac{\beta}{(1-\alpha)(1-\lambda)} R^{1/2} dB_t , \tag{3.14}$$

as N goes to $+\infty$ and a goes to 0, respectively, where $R = E\eta_n\eta_n'$ and $\{B_t\}$ is a standard Brownian motion.

This theorem can be shown by using the techniques developed in [2] and [3]. The stationary initial distribution of (3.14) is Gaussian distributed with mean zero and covariance $\frac{\beta}{(1-\alpha)(1-\lambda)^2} \int_0^\infty e^{g_x(0)t} R e^{g_x'(0)t} dt$.

REFERENCES

[1] H. J. Kushner and D. Clark, Stochastic Approximation Methods for Constrained and Unconstrained Systems, Springer-Verlag, New York, 1978.

[2] H. J. Kushner and H. Huang, "Rates of Convergence for Stochastic Approximation Type Algorithms", SIAM J. Control and Optimiz., 17 (1979), 607-617.

[3] H. J. Kushner and H. Huang, "Asymptotic Properties of Stochastic Approximations with Constant Coefficients", SIAM J. on Control and Optimiz., 19 (1981), 87-105.

[4] J. G. Proakis, "Advances in Equalization for Intersymbol Inference", Advances in Communication Systems: Theory and Applications, Vol. 4, Series Editor: A. V. Balakrishnan, Vol. Editor: A. J. Viterbi, Academic Press, New York, 1975.

[5] L. Ljung, "Analysis of Recursive Stochastic Algorithms", IEEE Trans. Automatic Control, AC-22, (1977), 551-575.

[6] P. Billingsley, Convergence of Probability Measures, Wiley, New York, 1968.

ON THE DECOUPLING OF LINEAR SYSTEMS INTO SINGLE INPUT-MULTIPLE OUTPUT SUBSYSTEMS

H. Y. KIM & E. Y. SHAPIRO
Lockheed California Company
Burbank, CA 91520

ABSTRACT

In this paper the necessary and sufficient conditions, for the decoupling of a linear time invariant system into single-input/multiple-output subsystems, are established. In addition a representation formula for the structure of the decoupling matrix pair {F,G} is developed.

I. INTRODUCTION

The decoupling problem of linear, time-invariant multivariable systems of the form

$$\dot{x} = Ax + Bu, \quad y=Cx \tag{1}$$

into subsystems by feedback control laws

$$u(t) = Fx + Gv \tag{2}$$

where $x \in R^n$, $y \in R^m$, u, $v \in R^q$, and A, B, C, F, G are constant matrices of appropriate size has a long history. Morgan [1] originally posed this problem, and obtained a partial solution when m=q. Later Falb and Wolovich [2] solved this special problem completely, establishing a necessary and sufficient condition for systems to be decoupled. Based on their results, Gilbert [3] derived representation formulas for the decoupling matrix pairs {F,G}.

Beginning in early 1970 Wonham and Morse [4] introduced a new approach to the general decoupling problem, contributing significantly in clarification of geometric concepts behind it.

Although the Wonham and Morse results are elegant, it is hard to implement in practical calculation. In 1975 Warren and Mitter [5] considered the problem of decoupling to single-input, multiple output (SIMO) subsystems along the line of Falb and Wolovich approach, and found an easily computable necessary condition.

The objective of this paper is to solve the SIMO decoupling problem completely by establishing a necessary and sufficient condition along the Gilbert's approach. A representation formula for the decoupling matrix pairs {F,G} is obtained which is similar to the one proved by Gilbert in his paper. Finally an example is given, clarifying the results obtained in this paper.

II. PROBLEM FORMULATION

Consider a system {A,B,C} given by (1), and control laws of (2). The transfer matrix H (s,F,G) of the system is then

$$H(s, F, G) = C(sI-A-BF)^{-1}BG \tag{3}$$

Suppose the output vector y is grouped into q subvectors

$$y_i = C_i x \tag{4}$$

where C_i is k_i x n, i = 1, 2, ..., q, with $\sum_{i=1}^{q} k_i = m$, and $C = (C_1', C_2', \ldots, C_q')'$. (The prime denote matrix transpose.) The decoupling problem we are interested in in this paper may be stated as follows: Find control laws u = Fx + Gv which yield the transfer function matrix H (s, F, G) in a block diagonal form

$$H(s,F,G) = \begin{bmatrix} t_1 & 0 & 0 & . & . & . & 0 \\ 0 & t_2 & 0 & . & . & . & 0 \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ 0 & 0 & 0 & . & . & . & t_q \end{bmatrix} \tag{5}$$

where t_i is k_i x 1 and none of the elements of t_i is identically zero. Thus we are interested in the input component v_i controlling output y_i completely without affecting the behavior of the remaining output y_j, $j \neq i$.

In [5] Warren and Mitter found a necessary condition for this problem. Before we state their results, it is helpful to settle upon notation.

For any integer r let $\underline{r}$ denote the set of integers {1,2,...,r}. Let k_i x n matrix $C_i = (C_{i1}', \ldots, C_{ik_i}')'$. Thus C_{ij} is the jth row of the matrix C_i. For each row C_{ir_i}, define the integer d_{ir_i} by $d_{ir_i} = \min\{n-1 \min\{j | C_{ir_i} A^j B \neq 0\}\}$ and for each $i \in \underline{q}$ the matrix D_i =

$((C_{i1} A^{d_{i1}} B)', \ldots, (C_{ik_i} A^{d_{ik_i}} B)')'$.

Then we may define $m \times q$ matrix D by stacking these $k_i \times q$ matrices D_i: $D = (D_1', \ldots, D_q')'$. Further, for each $i \in \underline{q}$, construct the $(m-k_i) \times q$ submatrix D_i^* of D by $D_i^* = (D_i', \ldots, D_{i-1}', D_{i+1}', \ldots, D_q')'$, and finally for each set of q integers $\{i_1, \ldots, i_q\}$ with $i_j \in \underline{k}_j$ $j \in \underline{q}$, we define the $q \times q$ submatrix $D_{i_1 \ldots i_q}$ of D by

$$D_{i_1 \ldots i_q} = \begin{bmatrix} C_{1i_1} A^{d_{1i_1}} B \\ \vdots \\ C_{qi_q} A^{d_{qi_q}} B \end{bmatrix} \tag{6}$$

With this notation the main result of Warren and Mitter may be stated as

THEOREM 1

A linear system may be decoupled only if

i) For every set $(i_1, \ldots, i_q)$ with $i_j \in \underline{k}_j$, $j \in \underline{q}$, the submatrix $D_{i_1 \ldots i_q}$ has rank q,

ii) For every $i \in \underline{q}$ the submatrix D_i^* has rank $q-1$.

Their proof is quite complicated. We present here a simple proof of their theorem. In the proof we use the following facts proved by Gilbert [3]. Denote the (i, j) th row of the transfer matrix $H(s,F,G)$ corresponding to the row vector C_{ij} of the matrix C by $H_{ij}(s, F, G)$ and define an integer $d_{ij}(s, F, G)$ and a row vector $D_{ij}(s, F, G)$ by the integer r such that $\lim_{s \to \infty} s^{r+1} H_{ij}(s, F, G)$ is nonzero and finite and by $\lim_{s \to \infty} s^{r+1} H_{ij}(s, F, G)$ respectively if $H_{ij}(s, F, G) \neq 0$. Otherwise, $d_{ij}(s, F, G) = n - 1$ and $D_{ij}(s, F, G) = 0$. It was proved that $d_{ij}(s, F, G)$ and $D_{ij}(s, F, G)$ thus defined are identical respectively to d_{ij} and $D_{ij}G$ where D_{ij} is the jth row of D_i provided that $H_{ij}(s, F, G) \neq 0$.

Suppose a matrix pair $\{F,G\}$ decouples the system. Thus the transfer matrix $H(s, F, G)$ is in the form of (5), which shows that $H_{ij}(s,F,G) \neq 0$ for all $j \in \underline{k}_i$, $i \in \underline{q}$. From this definition of $D_{ij}(s, F, G)$ it is clear that DG is also in the form of (5) where, of course, all nonzero entries in the column vector t_i, $i = 1, 2, \ldots, q$, are independent of s. As $D_{i_1 \ldots i_q} G$ is obtained from DG by deleting all rows except (r, i_r) th rows, $r \in \underline{q}$, it is obvious that $D_{i_1 \ldots i_q} G$ is a nonsingular diagonal matrix, and hence $D_{i_1 \ldots i_q}$ is a nonsingular matrix since G is nonsingular. Similarly it is easy to see $D_i^* G$ is a block diagonal matrix if the ith column is deleted from it, which tells that $D_i^* G$ is of rank $(q-1)$, and hence the rank of D_i^* is $q-1$, too.

III. SINGLE-INPUT, SINGLE-OUTPUT DECOUPLING PROBLEM

If $m=q$, the SIMO decoupling problem reduces to the single-input, single-output (SISO) problem. Then $k_i = 1$, $i \in \underline{q}$, whence matrix C_i is a row vector. We shall use C_i for this row vector and d_i for d_{i1} in this section. Furthermore the matrix D is now a $m \times m$ square matrix. Using this matrix Falb and Wolovich [2] showed that the system can be decoupled if and only if D is nonsingular.

Based on this result, Gilbert [3] found representation formulas for the decoupling matrix pairs $\{F,G\}$. As our results heavily depend on his theory, a brief explanation of his results follows. We shall assume in this paper that the system is completely controllable.

Define a new system $\bar{S} = \{\bar{A}, \bar{B}, \bar{C}\}$ where $\bar{A} = A - BD^{-1} A^*$, $\bar{B} = BD^{-1}$, $\bar{C} = C$, $A^* = (C_i A^{d_i+1})$. (This is called an integrator decoupled system in [3]. See also Theorem 1 of [2].) It is easy to show $d_i = \bar{d}_i$ where the definition of $\bar{d}_i$ for $\bar{S}$ is analogous to the definition of d_i for S. Let

$$W_i = \{\eta \in R^n \mid \eta \bar{A}^j \bar{B}_k = 0,\ j = 0, 1, \ldots, n - 1;\ k \neq i\} \tag{7}$$

where $\bar{B}_k$ is the kth column of $\bar{B}$. Then the following lemma is known [3].

LEMMA 1:

The subspace W_i has the properties:

- i) W_i is $\bar{A}$-invariant, i.e., $W_i \bar{A} \subset W_i$
- ii) $W_i \cap W_j = \{0\}$, $i \neq j$,
- iii) $C_i, C_i \bar{A}, \ldots, C_i \bar{A}^{d_i}$ are linearly independent vectors belonging to W_i.
- iv) $W_i \cap \{W_1 + \ldots + W_{i-1} + W_{i+1} + \ldots + W_m\} = \{0\}$

As $C_i, C_i \bar{A}, \ldots, C_i \bar{A}^{d_i}$ are linearly independent and belong to W_i, they can be extended to form a basis of W_i: $\{C_i, C_i \bar{A}, \ldots, C_i \bar{A}^{d_i}, q_1^i, \ldots, q_{r_i}^i\}$ is a basis of W_i, where $p_i = d_i + 1 + r_i$ is the dimension of the subspace W_i. For $i = 1, \ldots, m$, form a $p_i \times n$ matrix Q_i rows of which consist of the above basis of the subspace W_i. Then because of (iv) of Lemma 1 there exists a $(p_{m+1} \times n)$ matrix Q_{m+1}, where $p_{m+1} = n - (p_1 + ,,,+ p_m)$, such that $Q = (Q_1', \ldots, Q_m', Q_{m+1}')'$ is a nonsingular matrix. With this matrix the set of decoupling pairs $\{F,G\}$ is characterized by

THEOREM 2:

A matrix pair $\{F,G\}$ decouples the system if and only if

$$F = -D^{-1}A^* + D^{-1} \begin{bmatrix} \theta_1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & \theta_2 & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & \theta_m & 0 \end{bmatrix} Q \tag{8}$$

and

$$G = D^{-1} \operatorname{diag} (\lambda_1, \ldots, \lambda_m),\ \lambda_i \neq 0,\ i = 1, 2, \ldots, m, \tag{9}$$

where θ_i is $1 \times p_i$.

REMARK:

Although $p_i \times n$ matrix Q_i, $i = 1, 2, \ldots, m$, has $C_i, C_i \bar{A}, \ldots, C_i \bar{A}^{d_i}$ as the first $(d_i + 1)$ row vectors, it is obvious that any basis of W_i can be used in construction of Q_i for the representatin (8) for the matrix F.

The subspace W_i can be defined, as the following lemma shows, a little differently, and this will be continuously used in the sequel.

LEMMA 2:

$\eta \in W_i$ if and only if $\eta(A - \bar{B}_i A_i^*)^j \bar{B}_k = 0, \forall j,\ k \neq i$, where A_i^* is the ith row of A^*.

PROOF:

By definition if $\eta \in W_i$, $\eta(A-\bar{B}A^*)^j \bar{B}_k = 0, \forall j,\ k \neq i$. But $\eta(A-\bar{B}A^*)^j = \eta(A-\bar{B}A^*)^{j-1}(A-\bar{B}_iA_i^*) = \ldots = (A-\bar{B}_iA_i^*)^j$. Hence $\eta(A-\bar{B}_iA_i^*)^j \bar{B}_k = 0, \forall j,\ k \neq i$. Conversely if $\eta(A-\bar{B}_iA_i^*)^r \bar{B}_k = 0$ for all r and $k \neq i$, then $\eta(A-\bar{B}_iA_i^*)^j = \eta(A-\bar{B}_iA_i^*)^{j-1} (A-\bar{B}A^*) = \ldots = \eta(A-\bar{B}A^*)^j$, and hence $\eta(A-\bar{B}A^*)^j \bar{B}_k = 0$.

IV. MAIN RESULTS

As two conditions of Theorem 1 are necessary for SIMO decoupling we assume in the sequel that these conditions are satisfied. To proceed our investigation we need the following notation. Let

$$\tilde{C} = (C_{1i_1}', \ldots, C_{qi_q}')' \tag{10}$$

Since the system $\{A, B, \tilde{C}\}$ provides a SISO decoupling we may define the integrator decoupled system and subspace W_i. We shall denote the matrices $\bar{A}$, $\bar{B}$, A^*, and the subspace W_i by

$\bar{A}_{i_1 \ldots i_q}$, $\bar{B}^{i_1 \ldots i_q}$, $A^*_{i_1 \ldots i_q}$, $W^{i_1 \ldots i_q}$: $\bar{B}^{i_1 \ldots i_q} = BD^{-1}_{i_1 \ldots i_q}$, $A^*_{i_1 \ldots i_q} = \left[\left(C_{1i_i} A^{d_{1i_1}+1} \right)', \right.$

$\ldots, \left(C_{qi_q} A^{d_{qi_q}+1} \right)' \Big]'$, $\bar{A}_{i_1 \ldots i_q} = A - \bar{B}^{i_1 \ldots i_q} A^*_{i_1 \ldots i_q}$, $W_i^{i_1 \ldots i_q} = \{ \eta \in R^n \mid \eta \bar{A}^j_{i_1 \ldots i_q} \bar{B}_k^{i_1 \ldots i_q} = 0, \forall\, j, k \neq i \}$ where $\bar{B}_k^{i_1 \ldots i_q}$ is the kth column of the matrix $\bar{B}^{i_1 \ldots i_q}$.

Suppose $k_j \geqslant 2$ for some $j \in \underline{q}$. Then we have

LEMMA 3:

Let $r, p \in \underline{k}_j$. Then the rth and pth rows of the matrix D_j are parallel.

PROOF:

Take j = 1 for convenience. As two matrices $D_{i_1 i_2, , , i_q}$, $i_1 = p, r$ are of full rank, the first row of $D_{pi_2 \ldots i_q}$ can be expressed in terms of the rows of $D_{ri_2 \ldots i_q}$. If above two vectors are not parallel, q row vectors consisting of the first row vector of each of $D_{i_1, \ldots i_q}$, $i_1 = p, r$, and a suitable (q - 2) row vectors from the remaining (q-1) vectors of $D_{ri_2, , , i_q}$ are linearly independent since any q-dimensional vectors can be linearly expressed in terms of these q vectors. Suppose jth row ($j \neq 1$) of $D_{ri_2 \ldots i_q}$ be dropped in choosing the above (q - 2) vectors to form a set of independent vectors. Then D_j^* has rank q which is impossible because of the condition (ii) of Theorem 1. This proves the lemma.

This lemma tells that the rank of D_i, i = 1, ...q, is one. From this it is easy to see that the column vectors of $D^{-1}_{i_1 \ldots i_q}$ are parallel to the corresponding column vectors of $D^{-1}_{j_1 \ldots j_q}$. Consider $W_1^{i_1 \ldots i_q}$ for example. For a fixed $i_1 \in \underline{k}_1$, there appear $k_2 x \ldots x k_q$ such subspaces since different sets of $i_2, \ldots, i_q$, $i_j \in \underline{k}_j$, j = 2, ..., q, might yield different subspaces W_1. The following proposition shows that W_1 does not depend on the choice of $i_j \in \underline{k}_j$, $j \neq 1$, but only on $i_1 \in \underline{k}_1$. More generally we have

PROPOSITION 1:

If $i_r = j_r$, $W_r^{i_1 \ldots i_q} = W_r^{j_1 \ldots j_q}$

PROOF:

First observe that the columns of $\bar{B}^{i_1 \ldots i_q}$ are parallel to the corresponding columns of $\bar{B}^{j_1 \ldots j_q}$ as $D^{-1}_{i_1, \ldots i_q}$ and $D^{-1}_{j_1, \ldots j_q}$ have a similar relation. Furthermore rth column of $D^{-1}_{i_1, \ldots i_q}$ and $D^{-1}_{j_1 \ldots j_q}$ are identical as $i_r = j_r$. We know $\eta \in W_r^{i_1 \ldots i_q}$ if and only if $\eta (A - \bar{B}_r^{i_1 \ldots i_q} A^*_r)^j \bar{B}_k = 0, \forall\, j, k \neq r$, where A^*_r is the rth row of $A^*_{i_1 \ldots i_q}$, which is identical to the rth row of $A^*_{j_1 \ldots j_q}$. Therefore $\eta \in W_r^{i_1 \ldots i_q}$ if and only if $\eta \in W^{j_i \ldots j_q}$ if $i_r = j_r$. This completes the proof of the proposition.

The number of subspaces W we deal with reduces considerably. For example, if $k_j = k$, $j \in \underline{q}$ then instead of q k^q such subspaces we need to find only qk subspaces e.g. $W_i^{11 \ldots 1}, \ldots, W_i^{k \ldots k}$, i = 1,...,q.

Because of this proposition we may write

$$W_r^{i_1 \ldots i_q} = W_r^{i_r} \tag{11}$$

That is, as long as i_r in $\underline{k}_r$ is fixed W_r is uniquely determined. Using this notation the following characterization of the SIMO decoupling problem is possible.

PROPOSITION 2:

The system {A, B, C} may be decoupled if and only if

$$\bigcap_{j \in \underline{k}_r} (-a_{rj} \; A^*_{rj} + W^j_r) \neq \emptyset, \; r = 1, 2, \ldots, q. \tag{12}$$

where

$$A^*_{rj} = C_{rj} \; A^{d_{rj}+1} \text{ and } a_{rj} \text{ are from the relation}$$

$$C_{rj} \; A^{d_{rj}} B = \frac{1}{a_{rj}} \; C_{r1} \; A^{d_{r1}} \; B, \; j \in \underline{k}_r$$

PROOF

Suppose there exists a matrix pair {F, G} which block-diagonalizes the transfer matrix H (s, F, G). Then this pair must decouple the system {A, B, $\tilde{C}$} where $\tilde{C}$ is the matrix given in (10). Since this is a SISO decoupling problem Gilbert's theorem tells that F and G must be given by

$$D_{i_1 \ldots i_q} \quad F = -A_{i_1 \ldots i_q} + \begin{bmatrix} \theta_1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & \theta_2 & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & \theta_q & 0 \end{bmatrix} Q \tag{14}$$

Take $i_j = 1$, $j = 1, \ldots, q$, and consider the first row of (14). Let $f_1, \ldots, f_q$ be the row vectors of F and $(d_{11} \ldots d_{1q})$ be the first row of the matrix $D_{11 \ldots 1}$. Then

$$\sum_{j=1}^{q} d_{ij} \; f_j \; \in \; -A^*_{11} \; + W^1_1 \tag{15}$$

Suppose $k_1 > 1$ and take $i_1 = 2$, $i_j = 1$, $j = 2, \ldots, q$. As the first row of $D_{21 \ldots 1}$ is the first row of $D_{1 \ldots 1}$ multiplied by $(1/a_{12})$, we get

$$\frac{1}{a_{12}} \sum_{j=1}^{q} \; d_{ij} \; f_j \; \in -A^*_{12} \; + W^2_1 \tag{16}$$

Note $W^2_1 = W^{211 \ldots 1}_1$ in our old notation. (See Proposition 1). (15) and (16) tell that

$$\sum_{j=1}^{q} \; d_{ij} \; f_j \; \in \; (-A^*_{11} + W^1_1) \cap (-a_{12} \; A^*_{12} + W^2_1) \tag{17}$$

We can apply this argument to show that the condition (12) is necessary for decoupling. Conversely suppose that (12) is satisfied. Pick x_r from the set of the left hand side of (12) for $r = 1, 2, \ldots, q$. Define $F = D^{-1}_{1 \ldots 1} \; (x'_1, \ldots, x'_q)'$. Suppose this matrix F does not decouple the system. The second row, say, of the transfer matrix is not in the form of $(a, 0, \ldots, 0)$, $a \neq 0$ (assuming that $k_1 > 1$). Then this matrix must not decouple the system {A, B, $\tilde{C}$} when $\tilde{C}$ is the matrix given in (10) with $i_1 = 2$, $i_j = 1$, $j = 2, \ldots, q$. But this is impossible by Gilbert's theorem since $D_{21 \ldots 1}F$ is in the form of (14). This proves the sufficiency of the condition.

REMARK:

In the proof of the proposition we did not deal with G matrix. We shall drop it from our discussion compeltely with the agreement that G has the form of (9) with D replaced by $D_{1 \ldots 1}$. Note that $D^{-1}_{i_1 \ldots i_q} = D^{-1}_{1 \ldots 1} \Lambda$ where Λ is a nonsingular diagonal matrix.

Proposition 2 does not reveal much about the decoupling problem. However, we obtain from this an interesting characterization of the problem given by

THEOREM 3:

The system $\{A, B, C\}$ may be decoupled if and only if $W_r^{i_r}$ are identical for all $i_r \in \underline{k}_r$. That is, $W_r^{i_r}$ depends on r but not on i_r.

The proof of the theorem is provided in two steps each of which is presented as a lemma. We shall prove these lemmas for $r = 1$ assuming $k_1 > 1$. We assume that W_1^r, $r = 1,\ldots,k_1$ are constructed using some set $\{i_2,\ldots,i_q\}$, $i_j \in \underline{k}_j$. That is, $\tilde{C}$ of (10) is taken to be $\tilde{C} = (C'_{1r}, C'_{2i_2},\ldots,C'_{qi_q})'$.

As these subspaces W_1^r do not depend on particular i_j, $j \neq 1$, we may take $i_j = 1$. For convenience the subscript 1 will be supressed. Thus $\underline{k} = \underline{k}_1$, $a_j = a_{1j}$, $W^r = W_1^r$, $A_j^* = A_{1j}^*$. With this notation we have

LEMMA 4:

$$\bigcap_{r \in \underline{k}} (-a_r A_r^* + W^r) \neq \emptyset \text{ if and only if } (a_i A_i^* - a_j A_j^*) \in \bigcap_{r \in \underline{k}} W^r, \quad i,j \in \underline{k}$$

PROOF:

First we shall prove by induction that

$$(-a_i A_1^* + W^i) \cap (-a_j A_j^* + W^j) \neq \emptyset \tag{18}$$

if and only if

$$(a_i A_i^* - a_j A_j^*) \in W^i \cap W^j \tag{19}$$

for any $i, j \in \underline{k}$. Let x be a vector in the intersection of the two linear varieties of (18). As $x + a_r A_r^* \in W^r$, $r = i, j$,

$$(x + a_r A_r^*)(A - a_r \overline{B}_1 A_r^*)^t \, \overline{B}_k = 0, \quad \forall\, t,\ k \neq 1 \tag{20}$$

where $\overline{B}_1$ is the first column of the matrix $\overline{B}^{1\ldots1} = BD_{1\ldots1}^{-1}$. Observe that $\overline{B}_k$, $k \neq 1$, is, except non-zero constant multiplication, independent of what $\tilde{C}$ matrix we pick in constructing W_1^r, $r = i, j$. From (20) with $t = 0$, $(a_i A_i^* - a_j A_j^*)\, \overline{B}_k = 0$, $k \neq 1$. Let $A_r = A - a_r \overline{B}_1 A_r^*$, $r = i,j$ for notational convenience, and assume $(A_i^t - A_j^t)\, \overline{B}_k = 0$ and $(a_i A^*_i - a_j A_j^*)\, A_r^t\, \overline{B}_k = 0$, $r = i, j$, for $t = 0, 1,\ldots,p-1$, $k \neq 1$. Then as $A_i^p - A_j^p = A_i^{p-1}(A_i - A_j) + \ldots + (A_i - A_j) A_j^{p-1}$ and $A_i - A_j = -\overline{B}_1 (a_i A_i^* - a_j A_j^*)$, $(A_i^p - A_j^p)\, \overline{B}_k = 0$ by the induction hypothesis. Take $t=p$ in (20) and subtract one from another. Then, because of the last equation, we have

$$(a_i A_1^* A_i^p - a_j A_j^* A_j^p)\, \overline{B}_k = 0. \tag{21}$$

As

$$A_j^p = (A_i - (A_i - A_j))^p = A_i^p + E \tag{22}$$

where every term in E has at the tail factors in the form of $(a_i A_i^* - a_j A_j^*)\, A_i^r$, $r<p$, (21) becomes $(a_i A_i^* - a_j A_j^*)\, A_i^p\, \overline{B}_k = 0$. Simarily we can show $(a_i A_i^* - a_j A_j^*)\, A_j^p\, \overline{B}_k = 0$. The induction is completed and we showed that $(a_i A_i^* - a_j A_j^*) \in W^i \cap W^j$. Suppose now $\bigcap_{j \in \underline{k}} (-a_j A_j^* + W^j) \neq \emptyset$, and let $i, j \in \underline{k}$. For any $t \in \underline{k}$, $a_i A_i^* - a_j A_j^* = (a_i A_i^* - a_t A_t^*) + (a_t A_t^* - a_j A_j^*)$. Since both terms in the right hand side are in W^t so is the left hand side. This shows $a_i A_i^* - a_j A_j^* \in \bigcap_{r \in \underline{k}} W^r$, $i, j \in \underline{k}$.

To prove the converse pick x_1 from $\bigcap_{j \in \underline{k}} W^j$. Then $x_1 - A_1^* + a_r A_r^* = x_r \in W^r$, $\forall\, r \in \underline{k}$ and $x_1 - A_1^* = x_r - a_r A_r^* \in -a_r A_r^* + W^r$. Since $x_1 \in W^1$, this shows $x^1 - A_1^* \in \bigcap_{r \in \underline{k}} (-a_r A_r^* + W^r)$ and the proof of the lemma is completed.

LEMMA 5:

Suppose $(a_iA_i^* - a_jA_j^*) \in \bigcap_{r\in\underline{k}} W^r, \forall\, i, j\in\underline{k}$. Then W^r are identical for all $r\in\underline{k}$. Conversely if W^r are identical for $r\in\underline{k}$, then $(a_iA_i^* - a_jA_j^*) \in W$, $i, j\in\underline{k}$ where W is the subspaces W^r, $r\in\underline{k}$ which are identical.

PROOF:

As in the previous lemma we use the notation $A_r = A - a_r\bar{B}_1A_r^*$, $r\in\underline{k}$. Let $\eta\in W^i$. Then it suffices to show $\eta\in W^j$, $j\in\underline{k}$. Since $\eta\in W^i$, $\eta A_i^r\bar{B}_k = 0, \forall\, r, k \neq 1$. As $A_j = A_i + \bar{B}_1(a_iA_i^* - a_j A_j^*)$, $\eta A_j^r \bar{B}_k = 0, \forall\, r, k \neq 1$, since by assumption $(a_iA_i^* - a_jA_j^*) \in W^i$.
Conversely let $\eta\in W$. Then for any $i, j\in\underline{k}$.

$$\eta A_r^t \bar{B}_k = 0, \forall\, t, k \neq 1, r = i, j. \tag{23}$$

Take $t = d_i + 1$ and $\eta = C_i$ where $d_i = d_{1i}$ and $C_i = C_{1i}$. Since $C_i \in W^i$, $C_i A_i^{d_i+1} \bar{B}_k =$ $(C_i A^{d_i+1} - a_iC_i A^{d_i} \bar{B}_1A_i^*)\bar{B}_k = 0$ as $C_i A^rB = 0$ for $r<d$.
As $C_i \in W^j$, $C_i A_j^{d_i+1} \bar{B}_k = (C_i A^{d_i+1} - a_j C_i A^{d_i}\bar{B}_1 A_j^*)\bar{B}_k = 0$.

From these two equations we get $(a_iA_i^* - a_jA_j^*)\bar{B}_k = 0$, $k \neq 1$, since $C_i A^{d_i} \bar{B}_1 \neq 0$. Assume $(a_iA_i^* - a_jA_j^*) A_i^r \bar{B}_k = 0$, $k \neq 1$, for $r = 0, 1,\ldots,p-1$. Take $t = d_i+1+p$ and $\eta = C_i$ in (23). As in (22) $A_j^p \bar{B}_k = A_i^p \bar{B}_k$ by the induction hypothesis. Thus we have $C_i A_i^{d_i+1} A_i^p \bar{B}_k = 0$ and $C_i A_j^{d_i+1} A_i^p \bar{B}_k = 0$. From these two equations $(C_i A_i^{d_i+1} - C_i A_j^{d_i+1}) A_i^p \bar{B}_k =$ $-C_i A^{d_i} \bar{B}_1 (a_iA_i^* - a_jA_j^*) A_i^p \bar{B}_k = 0$ which completes the induction and the proof of the lemma.
The proof of Theorem 3 is obtained by combining above two lemmas together.

The following corollary is an easy consequence of Theorem 3.

COROLLARY:

If the system is decoupled to SIMO subsystems, then for each $r\in\underline{q}$, $\{-a_{rj} A_{rj}^* + W_r^j\}$ are identical for all $j\in\underline{k}_r$.

PROOF:

As the system is SIMO decouplable, W_r^j does not depend on j. Let $W_r^j = W_r$. By Lemma 5 $a_{ri} A_{ri}^* - a_{rj} A_{rj}^* \in W_r$. Hence $-a_{rj} A_{rj}^* + W_r \subset -a_{ri} A_{ri}^* + W_r$. This proves the corollary.
The following theorem provides representation formulas for the decoupling matrix pairs.

THEOREM 4:

Suppose $\{F, G\}$ decouples the system. Then F is given by

$$D_{1\ldots1} F = - A_{1\ldots1}^* + \begin{bmatrix} \theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \theta_2 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & & \theta_q & 0 \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_{q+1} \end{bmatrix} \tag{24}$$

and

$$G = D_{1\ldots1}^{-1} \operatorname{diag}(\lambda_1,\ldots,\lambda_q),\ \lambda_i \neq 0,\ i = 1, 2, \ldots, q, \tag{25}$$

where $(p_i \times n)$ matrix Q_i, $i = 1, 2, \ldots, q$, forms a basis of W_i and θ_i is $(1 \times p_i)$, (p_i is the dimension of the subspace W_i) and Q_{q+1} is a basis of W_{q+1} which corresponds to W_{m+1} of Theorem 2.

PROOF:

If $\{F, G\}$ decouples the system, it clearly decouples the system $\{A, B, \tilde{C}\}$ into SISO subsystems where $\tilde{C}$ is given by (10) with $i_j = 1$, $j \in \underline{q}$. By Gilbert's theorem F and G must be in the form of (24) and (25) respectively. The converse is proved in the same way as in the proof of Proposition 2. For the representation formula (25) of the matrix G see the remark after Proposition 2.

REMARK:

If we use $D_{i_1 \ldots i_q}$ in place of $D_{1 \ldots 1}$, then $A^*_{1 \ldots 1}$ must be replaced by $A^*_{i_1 \ldots i_q}$. It is obvious from the theory developed above that they are identical.

V. EXAMPLE

EXAMPLE (Warren and Mitter [5]):

Consider a system $\{A, B, C\}$ represented by

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad C_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}^T \quad C_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}^T \tag{26}$$

where the a_{ij}'s are not yet specified. It is easy to see all the $d_{ij} = 0$ and

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

Thus the necessary conditions for decoupling of Theorem 1 are met.

Let a_i, $i = 1, 2, 3, 4$, be the ith row of the matrix A. Then $A^* = CA = (a_1' + a_2', a_3' + a_4', a_4')'$ from which we get $\bar{A}_{11} = (a_1', -a_1', a_3', -a_3')'$ and $\bar{A}_{12} = (a_1', -a_1', a_3', 0)'$. It is easy to see then that $x = (x_1, x_2, x_3, x_4) \in W_2^{11}$ (we shall use the original notation in this section) if and only if $x_2 = 0$, $a_{12} x_1 + a_{32} (x_3 - x_4) = 0$, $\alpha x_1 + \beta(x_3 - x_4) = 0$, $\gamma x_1 + \delta(x_3 - x_4) = 0$, where α, β, γ, δ, are some constants which may be equal to zero, and $x \in W_2^{12}$ if and only if $x_2 = 0$, $a_{12} x_1 + a_{32} x_3 = 0$, $\alpha' x_1 + \beta' x_3 = 0$, $\gamma' x_1 + \delta' x_3 = 0$ where α', β', γ', δ', are some constants. If $W_2^{11} = W_2^{12}$, from the second of the equations defining the subspaces W_2^{11} and W_2^{12}, we get $a_{32} x_4 = 0$. If $a_{32} \neq 0$, then $C_{21} \in W_2^{11}$ implies the system $\{\bar{A}_{11}, \bar{B}\}$ is not controllable, and hence $a_{32} = 0$. Under the assumption that $a_{32} = 0$, if $a_{12} = 0$ it is easy to see that $W_2^{11} = W_2^{12} = \{(a, 0, b, c)\}$ where a, b, c, are arbitrary numbers. Since $A^*_{21} = (a_{31} + a_{41}, a_{42}, a_{32} + a_{43}, a_{34} + a_{44})$ and $A^*_{22} = (a_{41}, a_{42}, a_{43}, a_{44})$ it is clear $(-A^*_{21} + W^{11}) \cap (-A^*_{22} + W_2^{12}) \neq \emptyset$ and $A^*_{21} - A^*_{22} \in W_2$ $(=W_2^{11} = W_2^{12})$. If $a_{12} \neq 0$ then $a_{31} = 0$ for W_2^{11} to be identical with W_2^{12} since otherwise $x \in W_2^{11}$ requires $x_3 = x_4$, while $x \in W_2^{12}$ causes $x_3 = 0$. The subspace $W_2 = W_2^{11} = W_2^{12}$ in this case is given by $W_2 = \{(0, 0, a, b)\}$. As $A^*_{21} = (a_{41}, a_{42}, a_{33} + a_{43}, a_{34} + a_{44})$ and $A^*_{22} = (a_{41}, a_{42}, a_{43}, a_{44})$, it is clear $(-A^*_{21} + W_2^{11}) \cap (-A^*_{22} + W_2^{12}) = \emptyset$ and $A^*_{21} - A^*_{22} \in W_2$.

Observe that for any a_{ij}'s $W_1^{11} = W_1^{12}$. Indeed as $\bar{A}_{11}^2 \bar{B}_2 + a_{34} \times \bar{A}_{11} \bar{B}_2 = \bar{A}_{12}^2 \bar{B}_2$ except the fourth component and $\bar{A}_{11}^3 \bar{3}_2 + a_{34} \times \bar{A}_{12}^2 \bar{B}_2 + (\bar{A}_{11}^2 \bar{B}_2)_3 \bar{A}_{11} \bar{B}_2 = \bar{A}_{12}^3 \bar{B}_2$ except the fourth component where $(A_{11}^2 \bar{B}_2)_3$ is the third component of the vector $\bar{A}_{11}^2 \bar{B}_2$, the defining equations for W_1^{11} and W_1^{12} are equivalent.

Finally it is not hard to show, by inverting (sI-A-BF), that the transfer matrix H(s, F, G) is in a desired form if we take

$$F = \left[\begin{array}{c:c:c:c} -a_{11} \; -a_{21} + f_1 & -a_{22} + f_1 & -a_{13} - a_{23} & -a_{14} \; -a_{24} \\ \hdashline g_1 & -a_{42} & g_2 & g_3 \end{array}\right]$$

and

$$G = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, \quad \lambda_i \neq 0,\ i = 1,\ 2$$

when $a_{32} = a_{12} = 0$, and

$$F = \left[\begin{array}{c|c|c|c} -a_{11}\ -a_{21} + f_1 & -a_{12}\ -a_{22} + f_1 & -a_{13}\ -a_{23} & -a_{14}\ -a_{24} \\ \hline -a_{14} & -a_{42} & g_1 & g_2 \end{array}\right]$$

and G as above when $a_{32} = a_{31} = 0$, $a_{12} \neq 0$, where f_1, g_1, g_2, and g_3 are arbitrary numbers. Observe that the matrices F and G above are consistant with (24) and (9) respectively.

VI. CONCLUSION

The problem of decoupling a linear time-invarient multivariable system into single-input, multiple-output subsystems was investigated, establishing a necessary and condition for the solvability of the problem. Also proved was a representation formula for the decoupling matrix pairs {F, G}.

REFERENCES

1. B. S. Morgan, "The Synthesis of Linear Multivariable Systems by State Variable Feedback," Proc. 1964 JACC, Stanford, Calif. pp. 468-472.

2. P. L. Falb and W. A. Wolovich, "Decoupling in the Design of Multivariable Control Systems," IEEE Trans. Automatic Control, AC-12 (1967) pp. 651-659.

3. E. G. Gilbert, "The Decoupling of Multivariable Systems by State Feedback," SIAM J. Control vol. 7, 1969, pp. 50-63.

4. W. M. Wonham and A. S. Morse, "Decoupling and Pole Assignment in Linear Multivariable Systems: A Geometric Approach," SIAM J. Control vol. 8, 1970, pp. 1-18.

5. M. E. Warren and S. K. Mitter, "A Necessary Condition for Decoupling Multivariable Systems," Int. J. Control, vol. 21, 1975, pp. 177-192.

On Decoupling of Multivariable Square Plants

Mansour Eslami
Dept. of Electrical Engineering
SUNY at Stony Brook
Stony Brook, NY 11794

Manikant D. Lodaya
Bendix Advanced Tech. Center
9140 Old Annapolis Road
Columbia, MD 21045

Abstract

This paper discusses when and how a multivariable square plant can be perfectly decoupled, with constant pre- and post-multiplier matrices K and L. The diagonalization or decoupling problem is addressed when G(s), the plant transfer function is given and when a state space description is given. Necessary conditions for perfect diagonalization are derived when G(s) has no special properties. Some limitation and practical aspects of perfect diagonalization are discussed.

1. Introduction

One of the basic and fundamental problems in the control of multi-input multi-output (MIMO) systems is how to handle interactions among various inputs and outputs, especially if these interactions are strong. One technique for handling square plants is the Multivariable Nyquist Array technique (References 1,2,3,4,5). In this technique, open loop compensation is designed so that the modified plant is diagonally dominant to such an extent that the closed loop system design reduces to a set of independent single loop design problems. The difficulty with this technique, however, is the development of efficient ways to generating pre- and post-multiplying gain or transfer matrices that can transform a given square plant into a diagonally dominant plant.

In this paper we look into several classes of linear square plants that, for all range of frequencies, can be completely diagonalized. Naturally, if this perfect diagonalization for a given system can be achieved, then it would give better control system performance. We do not consider the more complex use of dynamic compensators for diagonalization purposes, although conceptually this is possible. The main result of our work is the classification of square plants for which a perfect diagonalization can be carried out.

The organization of this paper is as follows. In Section 2, we discuss exact diagonalization when only G(s) is known. In Section 3, we present a diagonalization procedure when a state space description is given. Necessary conditions for exact diagonalization of general LGK are derived in Section 4, and some limitations and practical considerations are given in Section 5.

2. Exact Diagonalization, Given the Plant Transfer Function

Consider the following closed-loop system.

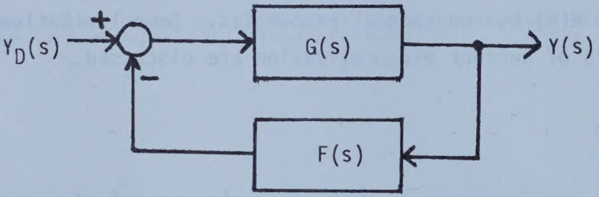


Figure 1

Where G(s) is an mxm plant matrix. We can expand G(s) as follows

$$G(s) = \frac{1}{\Delta(s)} \sum_{i=0}^{q} G_i s^i$$

Where $\Delta(s)$ is the common denominator polynomial of G(s), q is the power of the highest element of $\Delta(s)G(s)$ and

$$G_i = \frac{1}{i!} \frac{d^i}{ds^i} [\Delta(s)G(s)]\Big|_{s=0} \text{ for } i = 0,\ldots q$$

are constant matrices.

The mxm feedback matrix F(s) is generally chosen to be diagonal, and is designed such that the overall system meets certain performance requirements.

Let us use two compensators K and L as follows

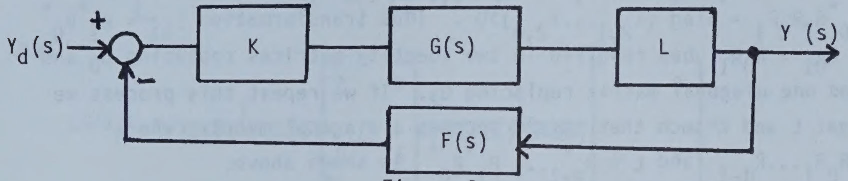


Figure 2

The question is:

Can LG(s)K be diagonalized?

Equivently, are looking for the case in which G_i, $i = 0,\dots, q$ can all be perfectly diagonalized. We recall the following standard result from matrix theory [Reference 6]. Let M and N be two nxn hermitian matrices (i.e., $M=M^*$ and $N=N^*$ here * stands for transpose conjugate). If M is positive definite, then there exists an nxn matrix R for which $R^*MR = I$ (Identity matrix), and $R^*NR = \text{diag}(\lambda_1,\dots\lambda_n)$. The λ_j's are generalized eignevalues satisfying $N\gamma_i = \lambda M\gamma_i$ where the generalized eigenvectors $\gamma_i \neq 0$ for $j = 1,\dots n$. If M and N are real, then a real matrix R, with columns γ_j, may be found such that $R^*MR = I$ and $R^*NR = \text{diag}(\lambda_1,\dots,\lambda_n)$, where λ_j's are the root of $\det(\lambda M-N) = 0$.

Going back to our problem we note that

$$G(s) = \frac{1}{\Delta(s)} \sum_{i=0}^{q} G_i \; s^i$$

and if there exist K and L such that LG(s)K is diagonal for all s then this K and L must diagonalize each and every G_i, $i = 0,\dots q$, as well as every summation of these constant matrices.

Theorem 1. Given $G(s) = \sum_{i=0}^{q} G_i s^i/\Delta(s)$ where G_i $i = 0,\dots q$ are symmetric, positive definite matrices for all i, with distinct eigenvalues for $i>1$ then we can find K and L matrices such that LG(s)K is diagonal for all s.

Proof: For $G(s) = \frac{1}{\Delta(s)} \sum_{i=0}^{q} G_i s^i$ with $G_i = G_i^T > 0$ for all i,

suppose (by Reference 6) we find R_0 such that $R_0^* G_0 R_0 = I$ and $R_0^* G_1 R_0 = \text{diag}\,(\lambda_{1,1} \cdots \lambda_{1,m})$ Since the eigenvalues of G_2 are distinct we can find R_1 such that $R_1^* I R_1 = I$, $R_1^* \,\text{diag}\,(\lambda_{1,1} \cdots \lambda_{1,m}) R_1 = I$ and $R_1^* R_0^* G_2 R_0 R_1 = \text{diag}\,(\lambda_{2,1} \cdots \lambda_{2,m}) > 0$. Thus transformation $L_{01} = R_1^* R_0^*$ and $K_{01} = R_0 R_1$ has resulted in two identity matrices replacing G_0 and G_1 and one diagonal matrix replacing G_2. If we repeat this process we can get L and K such that LG(s)K becomes a diagonal matrix, where $K = R_0 R_1 \cdots R_{q-1}$ and $L = R_{q-1}^* \cdots R_1^* R_0^*$ as shown above.

Example 1,

Consider a plant with Rosenbrock's transfer-function matrix (Reference 2)

$$G_p(s) = \frac{1}{(1+2s)(3+2s)} \begin{bmatrix} 4(s+1) & -2 \\ -2 & 4(s+1) \end{bmatrix}$$

We can show that based on our results there exist two constant compensators K and L which make this $G_p(s)$ perfectly diagonal.

Expanding $G_p(s)$ as:

$$G_p(s) = \frac{1}{(1+2s)(3+2s)} [G_o + G_1 s]$$

$$G_o = \begin{bmatrix} 4 & -2 \\ -2 & 4 \end{bmatrix} \quad G_1 = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

Here both G_o and G_1 are symmetric and positive definite, satisfying the requirements of our results.

Selecting

$$L = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \text{ and } K = \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & -1/2 \end{bmatrix}$$

based on eignevalue analysis, then

$$LG_0K = \begin{bmatrix} 2 & 0 \\ 0 & 6 \end{bmatrix} \text{ and } L\ G_1K = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

thus

$$LG(s)k = \frac{1}{(1+2s)(3+2s)}\begin{bmatrix} 2 & 0 \\ 0 & 6 \end{bmatrix} + \frac{1}{(1+2s)(3+2s)}\begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix} s$$

$$= \frac{1}{(1+2s)(3+2s)}\begin{bmatrix} 2+4s & 0 \\ 0 & 6+4s \end{bmatrix}$$

3. Perfect Diagonalization, given a state space form

Consider the plant state space form $\dot{X} = AX(t)+BU(t)$, $Y(t) = CX(t)$ where X is an nx1 plant state vector, u is the mx1 plant input vector and y is the px1 output vector. The pxm input-output transfer function matrix can be written as

$$G(s) = [C(sI-A)^{-1}B]$$

In general, if A has distinct engenvalues, we can find a matrix T such that TAT^{-1} is a diagonal matrix, we can write:

$$G(s) = C[T^{-1}T](sI-A)^{-1}(T^{-1}T)B$$

$$G(s) = CT^{-1}[T(sI-A)^{-1}T^{-1}]TB$$

$$= C'[T(sI-A)^{-1}T^{-1}]B'$$

where $C' = CT^{-1}$ and $B' = TB$

Furthermore expanding $(sI-A)^{-1}$ as follows (Reference 7):

$$(sI-A)^{-1} = (Is^{-1}+As^{-2}+A^2s^{-3}+A^3s^{-4}...)$$

we can write $T(sI-A)^{-1}T^{-1}$ as

$$T(sI-A)^{-1}T^{-1} = [Ts^{-1}T^{-1}+TAT^{-1}s^{-2}+TAAT^{-1}s^{-3}$$
$$TAAAT^{-1}s^{-4}\ldots]$$

Here

$$Ts^{-1}T^{-1} = s^{-1}TT^{-1}=s^{-1}$$

$$TAT^{-1}s^{-2}=Ds^{-2}$$

where D is a diagonal matrix

$$TAAT^{-1}s^{-3} = TAT^{-1}TAT^{-1} = D^2s^{-3}$$

Repeating this procedure for all other terms will give

$$T(sI-A)^{-1}T^{-1} = (Is^{-1}+Ds^{-2}+D^2s^{-3}+\ldots)$$

$$= (sI-D)^{-1}$$

a diagonal matrix.

Using this result we can re-write G(s) as

$$G(s) = C'[(sI-D)^{-1}]B'$$

Let C'^I be the left inverse of C' and B'^I be the right inverse of B' [Ref. 8] defined as follows:

$$C'^I = [C'^TC']^{-1} C'^T \text{ and } B'^I = B'^T[B'B'^T]^{-1}$$

Assuming C'^I and B'^I exist, we can write $C'^IC' = I$ and $B'B'^I = I$

Then we have

$$C'^IG(s)B'^I = (sI-D)^{-1}$$

which is a diagonal matrix.

The compensator matrices L and K are given by $L = C'^{I}$ and $K = B'^{I}$. In many cases we have p=m i.e., a square plant. If in addition we have $p=m\geq n$ then the existence of C'^{I} and B'^{I} is possible. Thus for a square plant with $P=m\geq n$ we may be able to diagonalize G(s) for all s provided that A is diagonalizable.

If G(s) were given as a transfer matrix which in general is a rectangular matrix then we could obtain a state space representation of it by any standard technique as given in (Ref. 7). Having a state space model we can then investigate the possibilities of diagonalization of G(s) as outlined above.

4. Necessary Conditions for Perfect Diagonalization of General LGK

Lets assume that K and L diagonalize $G(s) = \frac{1}{\Delta(s)} \sum_{i=o}^{q} G_i s^i$ for all s,

where G(s) does not necessarily have any of the properties assumed before. with this K and L we have

$LG(s)K = \tilde{G}(s)$, where $\tilde{G}(s)$ is diagonal

From the above equations we can see that K and L must diagonalize each and everyone of the G_i. Thus we also have

$LG_iK = \tilde{G}_i$, for $i = 0,\ldots,q$

where $\tilde{G}_i$ is a diagonal matrix with diagonal elements $\{\lambda_{i,j}\ j=1,\ldots m\}$

If we choose K and L nonsingular then rank of $\tilde{G}_i$ is the same as the rank of G_i and we can write $G_i = [L]^{-1}[\tilde{G}_i][K]^{-1}$

We observe that:

U = total number of unknowns

= $2m^2$ (m^2 unknown entries for each L and K)

+ $m(q+1)$ (total number of $\lambda_{i,j}$, for $i = 0,\ldots q$, $j=1,\ldots m$)

and

$$E = \text{the total number of equations}$$
$$= (q+1)m^2 \ (m^2 \text{ for each } G_i)$$

We will derive the necessary conditions for perfect diagonalization when

1. The number of unknowns are same as the number of equations
2. The number of unknowns are more than the number of equations, and
3. The number of unknowns are less than the number of equations.

When U = E

$$2m^2 + m(q+1) = (q+1)m^2$$

or

$$m = \frac{q+1}{q-1}$$

This is possible only when when q = 2 and m = 3 or q = 3 and m = 2

When U > E

$$2m^2+m(q+1)>(q+1)m^2$$

or

$$m< \frac{q+1}{q-1}$$

This is valid when

$q=2$ and $m=2$
or
$q \geq 3$ and $m=1$

When $U < E$

$$2m^2+m(q+1)<(q+1)m^2$$

or

$$\left(\frac{q+1}{q-1}\right) < m$$

Then the necessary conditions for diagonalization are

$q=2$ and $m>3$,

$q=3$ and $m>2$ or

$q\geq 4$ and $m>1$

5. Limitations and Practical Considerations

In this paper we have classified multivariable square plants for which perfect diagonalization is possible and proposed techniques for achieving the diagonalization. Our results at first glance may seem somewhat unrealistic. However, one has to realize the difficulty of approximation involved in the MNA design. If a given plant has the desired properties, it could be perfectly diagonalized, thus reducing the multi-loop system design problem to a set of independent single-loop design problems. If the plant does not satisfy all of the properties, then one may look into the possibilities of approximating the problem to meet these properties. We leave the justification for the approximation to the user. If the approximation fails then we can go back to the multivariable Nyquist Array technique.

REFERENCES

1. H. H. Rosenbrock, Computer-Aided Control System Design, New York: Academic Press, 1974.

.2 H. H. Rosenbrock, "Design of Multivariable Control Systems Using the Inverse Nyquist Array," Proc. IEE, Vol. 116, pp 1929, 1969.

3. N. Munro, "Conversational Model CAD of Control System", Proc. Institution of Electrical Engineers (IEE) International Conference on Computer-Aided Design, England, pp 418-431, 1972.

4. G. G. Leininger, "Diagonal Dominance for Multivariable Nyquist Array Methods Using Function Minimization," Automatica Vol. 15, No. 3, pp 339-346, 1979.

5. G. G. Leininger, "New Dominance Characteristics for the Multivariable Nyquist Array Method," Int. J. Control, Vol. 30, No. 3, pp 459-475, 1979.

6. J. N. Franklin, Matrix theory. Englewood Cliffs, N.J.: Prentice-Hall, 1968.

7. C. T. Chen, Introduction to Linear System Theory. New York: Holt, Reinhart and Winston, 19770.

8. B. Rust, W. R. Burrus and C. Schneeberger, "A Smiple Algorithm for Computing the Generalized Inverse of a Matrix," Association of Computing Machinary, Communication, Vol. 9. No. 5, pp 381-385 and 387, May 1966.

On the Selection of the Eigenvalues
to be Retained in a Reduced-Order Model

Guoliang Zhao*, P. Rozsa** and N.K. Sinha
Department of Electrical and Computer Engineering
McMaster University
Hamilton, Ontario, Canada

Abstract: A method is proposed for selecting the eigenvalues to be retained in an aggregated low-order model of a large-scale system. Based on the impulse-response energy concept, it is also applicable to multivariable systems.

I. Introduction

The methods of obtaining reduced-order models can be divided into two main classes: (i) those which retain the important eigenvalues of the original system [1-4], and (ii) those which select the reduced-order model in such a manner that an optimum approximation is made, without being concerned with the eigenvalues [5-8]. The aggregation method belongs to the first type, and may be regarded as the most general case of all methods which retain some eigenvalues of the original system. This is a projective reduction method and has several advantages, especially when it is combined with moments matching [9].

An important problem while using the first class of reduction methods is the selection of the eigenvalues which should be retained in the reduced-order model.

Consider a high-order, linear, time invariant system described by the following equations

$$\dot{x} = Ax + Bu \tag{1}$$

$$y = Cx \tag{2}$$

where $x \in R^n$, $u \in R^m$, and $y \in R^p$. It will be assumed that $n >> p$ and the system is asymptotically stable. In the processing of the aggregation method, it is generally convenient to transform the state vector so that the matrix A is in the block diagonal form, where the diagonal dominant eigenvalues, defined as those closest to the imaginary axis in the complex frequency plane, are retained in the reduced-order model, i.e. the rate of decay of the response is only considered. However, the rate of decay is not sufficient for calculating the system response, since the magnitude of the response caused by this eigenvalue has to be taken into account as well. Wilson, Fisher and Seborg [10] have pointed this out, but have not suggested how to take into account both of these aspects in the selection of the eigenvalues to be retained in the low-order model. Hutton and Friedland [6] have introduced the concept of impulse energy, which was shown by them as effective for single-input, single-output systems reduced by the Routh approximation method.

---

* On leave from the Harbin Shipbuilding Engineering Institute, The People's Republic of China.

** On leave from Technical University of Budapest, Hungary.

In this paper, we attempt to extend this concept to the multivariable case, and examine its applicability to the aggregation method.

II. Single-Input-Single-Output System

In this case, B and C in (1) and (2) are n x 1 matrices:

$$B = (b_1, b_2, \ldots, b_n)^T \tag{3}$$

$$C = (c_1, c_2, \ldots, c_n) \tag{4}$$

The transfer function of the system has the following form

$$H(s) = C(sI - A)^{-1} B \tag{5}$$

and the output of the system can be written as

$$y(t) = \mathcal{L}^{-1}[H(s) \cdot U(s)] \tag{6}$$

where $\mathcal{L}^{-1}[\cdot]$ denotes the inverse Laplace transform of $[\cdot]$. $U(s)$ is the Laplace transform of the input $u(t)$.

If the input is an impulse function, then the output of the system is of the form

$$y(t) = \mathcal{L}^{-1}[H(s)] = h(t) \tag{7}$$

where $h(t)$ is the impulse response of the system.

For the sake of simplicity, we assume that the eigenvalues of the system are distinct and real, i.e.

$$A = \begin{bmatrix} -\lambda_1 & 0 & \cdots & 0 \\ 0 & -\lambda_2 & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & --- & -\lambda_n \end{bmatrix} \tag{8}$$

Substituting (8), (3) and (4) into (5), we obtain

$$H(s) = \frac{c_1 b_1}{s + \lambda_1} + \frac{c_2 b_2}{s + \lambda_2} + \ldots + \frac{c_n b_n}{s + \lambda_n} \tag{9}$$

Hence,

$$h(t) = h_1(t) + h_2(t) + \ldots + h_n(t) \tag{10}$$

where

$$h_k(t) = {}^{-1}[H_k(s)] = \mathcal{L}^{-1}[\frac{c_k b_k}{s + \lambda_k}] = c_k b_k e^{-\lambda_k t}, \quad k = 1,2,3,\ldots,n \qquad (11)$$

The impulse response energy of the system is defined as follows

$$E = \int_o^\infty h^2(t) \qquad (12)$$

Substituting (10) into (12), we obtain

$$E = \sum_{k=1}^{n} \int_o^\infty h_k^2(t)\, dt + 2 \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} \int_o^\infty h_k(t)\, h_j(t)\, dt$$

$$= \sum_{k=1}^{n} E_k + 2 \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} E_{kj} \qquad (13)$$

where

$$E_k = \int_o^\infty h_k^2(t)\, dt = \frac{(b_k c_k)^2}{2\lambda_k} \qquad (14)$$

and

$$E_{kj} = \int_o^\infty h_k(t)\, h_j(t)\, dt \qquad (15)$$

It is obvious that E_k represents the contribution to the impulse response h(t) by the k^{th} eigenvalue λ_k individually.

It is known that the impulse response energy of the system can be represented in terms of Routh parameters α_i and β_i [6], i.e.

$$E = \int_o^\infty h^2(t)\, dt = \sum_{i=1}^{n} \frac{\beta_i^2}{2\alpha_i} \qquad (16)$$

The value of E_k for each component h_k reduces to equation (14) for a distinct real eigenvalue λ_k.

For the complex conjugate eigenvalues λ_k, $\lambda_{k+1} = -\xi \pm j\eta$, matrices A, B and C has in the following blocks

$$A = \begin{bmatrix} \ddots & & & \\ & -\xi & \eta & \\ & -\eta & -\xi & \\ & & & \ddots \end{bmatrix} \qquad (17)$$

$$B^T = (\text{---}\; b_k,\; b_{k+1}\; \text{---}) \tag{18}$$

$$C = (\;\text{---}\; c_k,\; c_{k+1}\; \text{---}) \tag{19}$$

It can be shown by using Routh parameters that the energy components of the impulse response E_k and E_{k+1} are of the following form:

$$E_k + E_{k+1} = \frac{1}{4\xi(\xi^2 + \eta^2)}[2\xi^2(b_k c_k + b_{k+1}c_{k+1})^2 + 2\xi\eta[c_k c_{k+1}(b_k^2 - b_{k+1}^2) + b_k b_{k+1}(c_{k+1}^2 - c_k^2)] + \eta^2(b_k^2 + b_{k+1}^2)(c_k^2 + c_{k+1}^2) \tag{20}$$

For the double real eigenvalue $-\lambda_k$ and A has in the Jordan normal blocks, while B, C are in the corresponding forms. Namely,

$$A = \begin{bmatrix} \ddots\; -\lambda_k & 1 \\ 0 & -\lambda_k \;\ddots \end{bmatrix} \tag{21}$$

$$B^T = (\text{---}\; b_k,\; b_{k+1}\; \text{---}) \tag{22}$$

$$C = (\text{---}\; c_k,\; c_{k+1}\; \text{---}) \tag{23}$$

Energy components of the impulse response E_k and E_{k+1} can be obtained by using Routh parameters as follows:

$$E_k + E_{k+1} = \frac{1}{2\lambda_k}[(b_k c_k + b_{k+1}c_{k+1})^2 + \frac{b_{k+1}c_k(b_k c_k + b_{k+1}c_{k+1})}{\lambda_k} + \frac{(b_{k+1}c_k)^2}{2\lambda_k^2}] \tag{24}$$

Therefore, from energy components E_k of the impulse response calculated according to (16), (20) and (24), respectively, we may select the larger ones and neglect the others, i.e. only those eigenvalues will be retained which correspond to the larger energy of the impulse response.

III. Multiple-Input-Multiple-Output System

Now the matrices B and C can be written as

$$B = \begin{bmatrix} b_{11} & b_{12} & \text{---} & b_{1m} \\ \vdots & & & \vdots \\ b_{n1} & b_{n2} & \text{---} & b_{nm} \end{bmatrix} \tag{25}$$

$$C = \begin{bmatrix} c_{11} & c_{12} & \text{---} & c_{1n} \\ \vdots & & & \vdots \\ c_{p1} & c_{p2} & \text{---} & c_{pn} \end{bmatrix} \tag{26}$$

We assume that the input u(t) consists of a set of impulse functions, i.e.

$$u(t) = \begin{bmatrix} 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} \delta(t) \tag{27}$$

where $\delta(t)$ is an impulse function. Hence equation (1) can be written as

$$\dot{x} = Ax + \tilde{B}\,\delta(t) \tag{28}$$

where $\tilde{B}$ is a column vector consisting of the sum of the columns of B and can be written as

$$\tilde{B}^T = (\tilde{b}_1, \tilde{b}_2, ---, \tilde{b}_n) \tag{29}$$

Assuming zero initial values, we obtain

$$x(t) = e^{At}\,\tilde{B} \tag{30}$$

and

$$y(t) = C\,e^{At}\,\tilde{B} \tag{31}$$

As above, let us define the impulse response energy of the system as

$$E = \int_0^\infty y^T\, y\, dt \tag{32}$$

Substituting (31) into (32), the impulse response energy of the system becomes

$$E = \int_0^\infty \tilde{B}^T\, e^{A^T t}\, C^T C\, e^{At}\, \tilde{B}\, dt \tag{33}$$

It can be shown that

$$E = \tilde{B}^T\, Q\, \tilde{B} \tag{34}$$

where $Q = [q_{ij}]$ is a symmetric positive semi-definite matrix of order n which satisfies the following equation

$$A^T Q + QA = -\,C^T\, C \tag{35}$$

If the eigenvalues of the matrix A are distinct and real, i.e. A has the form of (8), then equation (35) can easily be solved in explicit form. Namely, from (35), we get for each pair of indices i, j

$$-(\lambda_i + \lambda_j)q_{ij} = -\sum_{\ell=1}^{p} c_{\ell i} c_{\ell j}; \qquad i,\, j = 1,\, 2,\, \ldots,\, n \tag{36}$$

Hence,

$$q_{ij} = \frac{\sum_{\ell=1}^{p} c_{\ell i} c_{\ell j}}{\lambda_i + \lambda_j} \qquad i, j = 1, 2, \ldots, n \tag{37}$$

Substituting (37) into (34), the energy component E_k of the impulse response corresponding to the real distinct eigenvalue λ_k individually has the form of

$$E_k = \frac{\hat{b}_k^2 \sum_{\ell=1}^{p} c_{\ell k}^2}{2\lambda_k} \tag{38}$$

For the complex eigenvalues, i.e. $-\xi \pm j\eta$, the corresponding energy components E_k and E_{k+1} can be obtained as follows

$$\begin{aligned} E_k + E_{k+1} = \frac{1}{4\xi(\xi^2 + \eta^2)} \{ & 2\xi^2 [\tilde{b}_k^2 \sum_{\ell=1}^{p} c_{\ell k}^2 + 2\tilde{b}_k \tilde{b}_{k+1} \sum_{\ell=1}^{p} (c_{\ell,k+1} c_{\ell k}) \\ & + \tilde{b}_{k+1}^2 \sum_{\ell=1}^{p} c_{\ell,k+1}^2] + 2\xi\eta [(\tilde{b}_k^2 - \tilde{b}_{k+1}^2) \sum_{\ell=1}^{p} (c_{k+1,\ell} c_{\ell,k+1}) \\ & + \tilde{b}_k \tilde{b}_{k+1} (\sum_{\ell=1}^{p} c_{\ell,k+1}^2 - \sum_{\ell=1}^{p} c_{\ell k}^2)] \\ & + \eta^2 (\tilde{b}_k^2 + \tilde{b}_{k+1}^2) (\sum_{\ell=1}^{p} c_{\ell k}^2 + \sum_{\ell=1}^{p} c_{\ell,k+1}^2) \} \end{aligned} \tag{39}$$

For the double real eigenvalues, $-\lambda_k$, A is in the Jordan normal form, we obtain

$$\begin{aligned} E_k + E_{k+1} = \frac{1}{2\lambda_k} [& \tilde{b}_k^2 \sum_{\ell=1}^{p} c_{\ell k}^2 + 2\tilde{b}_k \tilde{b}_{k+1} \sum_{\ell=1}^{p} (c_{\ell k} c_{\ell,k+1}) + \tilde{b}_{k+1}^2 \sum_{\ell=1}^{p} c_{\ell,k+1}^2 \\ & + \frac{\tilde{b}_{k+1}^2 \sum_{\ell=1}^{p} (c_{\ell k} c_{\ell,k+1})}{\lambda_k} + \frac{\tilde{b}_k \tilde{b}_{k+1} \sum_{\ell=1}^{p} c_{\ell k}^2}{\lambda_k} + \frac{\tilde{b}_{k+1}^2 \sum_{\ell=1}^{p} c_{\ell k}^2}{2\lambda_k^2}] \end{aligned} \tag{40}$$

Equations (38), (39) and (40) can be used for calculating the energy components of the impulse response corresponding to each eigenvalue individually. Then we retain those eigenvalues in the reduced-order model which have larger energy components of the impulse response.

IV. Example

I. SISO System

Given a linear time-invariant system

$$\dot{x} = \begin{bmatrix} -0.5 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & 0 & -6 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} u$$

$$y = (0.2 \quad 1 \quad 20 \quad 20)\, x$$

The system has four distinct real eigenvalues at -0.5, -3, -5 and -6.

Therefore, the energy components of the impulse response caused by each eigenvalue individually are

$E_1 = 0.4$, $E_2 = 0.166$, $E_3 = 40$, $E_4 = 3.3$

Hence, $E_3 > E_4 > E_1 > E_2$. For a second-order reduced model, it is better to retain the eigenvalues at -5 and -6 rather than the eigenvalues at -0.5 and -3.

Using aggregation method with partial realization, we obtain two second-order reduced models for the original system: one retains the eigenvalue -5 and -6, the other retains the eigenvalues -0.5 and -3.

The first reduced model has the following form

$$\dot{z} = \begin{bmatrix} -5 & 0 \\ 0 & -6 \end{bmatrix} z + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u$$

$$y = (35.98 \quad 5.22)\, z$$

The second has the form of

$$\dot{z} = \begin{bmatrix} -0.5 & 0 \\ 0 & -3 \end{bmatrix} z + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u$$

$$y = (-3.4 \quad 44.6)\, z$$

Figure 1 shows the step response for the original system and for the reduced-order systems. It is obvious that the first reduced-order model is a better approximation.

II. MIMO System

Given a linear time-invariant system as follows

$$\dot{x} = \begin{bmatrix} -0.910 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -4.449 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -10.262 & 571.749 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -571.749 & -10.262 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -10.987 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -15.214 & 11.622 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -11.622 & -15.214 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -89.874 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -502.665 \end{bmatrix}$$

$$+ \begin{bmatrix} -4.336 \\ -3.691 \\ 10.140 \\ -1.612 \\ 16.629 \\ -242.476 \\ -14.261 \\ 13.672 \\ 82.187 \end{bmatrix} u$$

$$y = \begin{bmatrix} -0.422 & -0.736 & -0.00416 & 0.232 & -0.816 & -0.715 & 0.546 & -0.235 & -0.0806 \\ -0.00286 & 0.00136 & 0.117 & -0.393 & 0.00113 & -0.045 & 0.0685 & -0.0798 & -0.129 \end{bmatrix} x$$

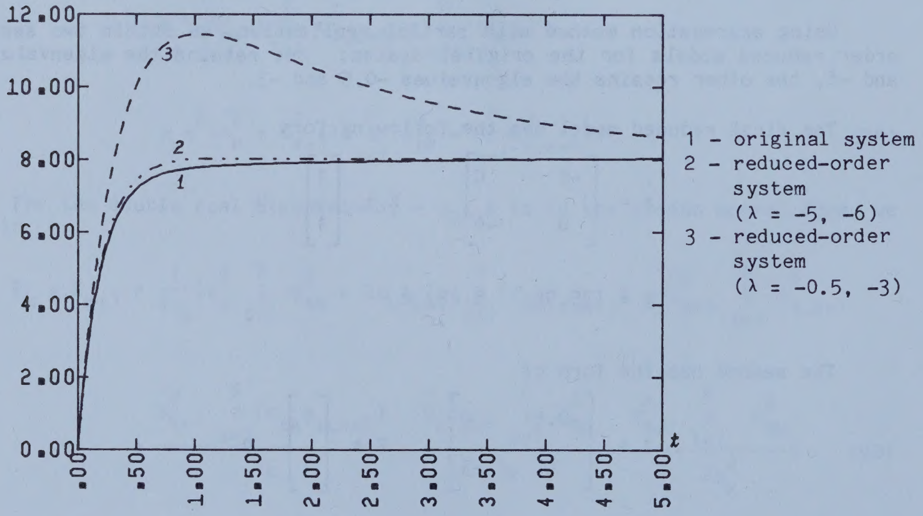


Fig. 1

The system has the following eigenvalues:

-0.910, -4.449, -10.262 ± j 571.749, -10.987, -15.214 ± j 11.262
-89.874, -502.665

The energy components of impulse response corresponding to each eigenvalue individually are

$$E_1 = 1.84,\ E_2 = 0.829,\ E_3 + E_4 = 0.559,\ E_5 = 8.35,\ E_6 + E_7 = 483.4$$

$$E_8 = 0.064,\ E_9 = 0.155$$

Hence, $(E_6 + E_7) > E_5 > E_1 > E_2 > (E_3 + E_4) > E_9 > E_8$.

Using aggregation with partial realization, we obtain several reduced-order models as follows:

1) Retaining eigenvalues -15.214 ± j 11.622 and -9.910

$$\dot{z} = \begin{bmatrix} -0.910 & 0 & 0 \\ 0 & -15.214 & 11.622 \\ 0 & -11.622 & -15.214 \end{bmatrix} z + \begin{bmatrix} -4.336 \\ -242.476 \\ -14.261 \end{bmatrix} u$$

$$y = \begin{bmatrix} -0.426 & -0.620 & 0.592 \\ -0.00285 & -0.0423 & 0.0693 \end{bmatrix} z$$

2) Retaining eigenvalues -0.910, -4.449 and -10.262 ± j 571.749

$$\dot{z} = \begin{bmatrix} -0.910 & 0 & 0 & 0 \\ 0 & -4.449 & 0 & 0 \\ 0 & 0 & -10.262 & 571.749 \\ 0 & 0 & -571.749 & -10.262 \end{bmatrix} z + \begin{bmatrix} -4.336 \\ -3.691 \\ 10.140 \\ -1.612 \end{bmatrix} u$$

$$y = \begin{bmatrix} -0.421 & -0.767 & -12005.9 & 1124.2 \\ -0.00279 & -0.049 & -1382.29 & 145.056 \end{bmatrix} z$$

3. Retaining eigenvalues -10.987, -15.214 ± j 11.622

$$\dot{z} = \begin{bmatrix} -10.987 & 0 & 0 \\ 0 & -15.214 & 11.622 \\ 0 & -11.622 & -15.214 \end{bmatrix} z + \begin{bmatrix} 16.629 \\ -242.476 \\ -14.261 \end{bmatrix} u$$

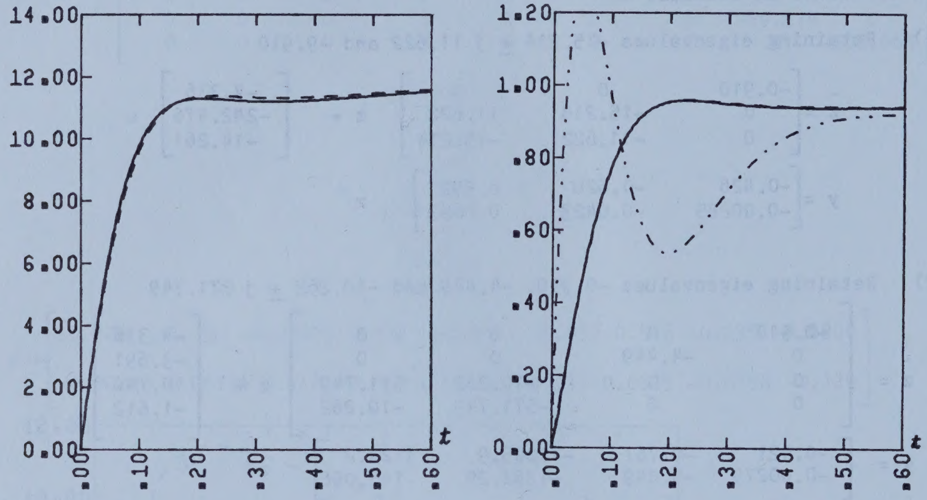
$$y = \begin{bmatrix} 434.083 & -0.045 & -90.907 \\ 2.903 & 0.038 & -0.543 \end{bmatrix} z$$

Figure 2 shows the step responses for the original system and for the three reduced-order systems. For output y_1, the response of only one reduced-order model is shown as the other two were oscillatory and not of scale. For output y_2, two of them have been shown for the same reason. The best reduced-order model is the one which retains the eigenvalues -0.91 and -15.214 ± j 11.622. It is easy to understand that the eigenvalues which are closer to the imaginary axis in the complex frequency plane have higher priority to be retained in the reduced-order model when the impulse energy components are also of large magnitude.

V. Conclusions

In most cases, the dominant eigenvalues, defined as those closest to the imaginary axis in the complex frequency plane, can be retained in the reduced-order model. It is better, however, to check each energy component of the impulse response before the reduction process.

It is obvious that when some of the energy components E_k of the impulse response are nearly equal, the eigenvalues closer to the imaginary axis in the complex frequency plane have higher priority.



(a) Output y_1

(b) Output y_2

Fig. 2

——— original system

- - - retaining $\lambda_1, \lambda_6, \lambda_7$

——— original system

- - - retaining $\lambda_1, \lambda_6, \lambda_7$

— - - — - - — retaining $\lambda_5, \lambda_6, \lambda_7$

The simplest model reduction method neglects some of the eigenvalues, (i.e. in the classical control theory). When doing so, the method described in this paper can be applied to decide the suitable eigenvalues to be retained.

An important disadvantage of the proposed method is that one must start with transforming the state equations so that the resulting A matrix is in the block diagonal form. This often requires a considerable amount of computation for a large-scale system. But it may be added that this computation is essential for all methods based on retaining the important eigenvalues of the system in the reduced-order model. The proposed method can be utilized for selecting the eigenvalues to be retained in a more logical manner.

REFERENCES

1. Davison, E.J., "A method for simplifying linear dynamic systems", IEEE Trans., AC-11, pp. 93-101, 1966.
2. Mitra, D., "The reduction of complexity of linear, time invariant dynamic systems", Proc. 4th IFAC Congress, 1969, Tech. Sect. 7, 1969.
3. Aoki, M., "Control of large scale dynamic systems by aggregation", IEEE Trans., AC-13, pp. 246-253, 1968.
4. Hickin, J. and Sinha, N.K., "Model reduction for linear multi-variable systems", IEEE Trans., AC-25, pp. 1121-1127, 1980.
5. Chen, C.F. and Shieh, L.S., "A novel approach to linear model simplifications", Int. J. Control, vol. 8, pp. 561-570, 1968.
6. Hutton, M.F. and Friedland, B., "Routh approximation for reducing order of linear time-invariant systems", IEEE Trans, AC-20, pp. 329-327, 1975.
7. Shamash, Y., "Stable reduced-order models using Padé-type approximations", IEEE Trans., vol. AC-19, p. 615, 1974.
8. Elliott, H. and Wolovich, W.A., "A frequency domain model reduction procedure", Automatica, vol. 16, pp. 167-177, 1980.
9. Elrazaz, Z. and Sinha, N.K., "A review of some model reduction techniques", Canadian Electrical Engineering Journal, vol. 6, no. 1, pp. 34-40, 1981.
10. Wilson, R.G., Fisher, D.G. and Seborg, D.E., "Model reduction and the design of reduced-order control laws". AIChE Journal, vol. 20, No. 6, pp. 1131-1140, 1974.

INSENSITIVE REDUCED-ORDER MODELS

MINH T. TRAN and MAHMOUD E. SAWAN
Electrical Engineering Dept., Wichita State Univ., Wichita, Ks. 67208

ABSTRACT

The problem of reducing the order of system with slow and fast modes is considered. A method is proposed for designing a state feedback gain which achieves a specified insensitivity of the closed loop trajectory.

SUMMARY

As shown in [1], a composite system can be described as

$$\begin{aligned} \dot{X} &= AX + BZ + EU_1 \\ \alpha\dot{Z} &= CX + DZ + FU_2 \end{aligned} \qquad \text{or} \qquad \dot{\overline{X}} = \overline{A}(\alpha)\overline{X} + \overline{B}U \tag{1}$$

where $X\epsilon R^{n_1}$, $Z\epsilon R^{n_2}$, $U_1\epsilon R^{m_1}$, $U_2\epsilon R^{m_2}$, $\overline{X}\epsilon R^{n_1+n_2}$, $U\epsilon R^{m_1+m_2}$, A, B, C, D, E, F, $\overline{A}$, $\overline{B}$ are matrices of appropriate dimension and α is a scalar which denotes the ratio of speed of slow and fast modes. The nominal system, which is obtained by letting $\alpha = \overline{\alpha} = 0$, is

$$\dot{X}_r = A_r(\overline{\alpha})X_r + B_r(\overline{\alpha})U_1 \tag{2}$$

where $A_r(\overline{\alpha}) = A - BD^{-1}C$ and $B_r(\overline{\alpha}) = E - BD^{-1}F$ (3)

The performance index at the nominal value along with the scalar Lagrange multiplier λ is

$$J(\overline{\alpha},\lambda) = \int_0^\infty\{\lambda[X_r^TQ_rX_r] + U_1^TR_rU_1\}dt \tag{4}$$

where Q_r and R_r are weighting matrices of appropriate dimension.

As described in [2] the closed loop response variation is limited to a prespecified extent if the following condition is met

$$\text{Max}[||P||_s] \leq \rho$$

where ρ is a positive scalar that specifies the allowable trajectory variation and $||P||_s$ is the maximum eigenvalue of the real, symmetric matrix P which satisfies the Lyapunov equation: $G^TP + PG + Q = 0$ with G is the closed loop model of the nominal system (2) and the performance index (4). It is found that by using this method, as λ increases (corresponding to increasing the cost), the maximum eigenvalue of P decreases which means that the sensitivity is improved.

REFERENCES

[1] J. H. Chow and P. V. Kokotovic, "A decomposition of near optimal regulators for systems with slow and fast modes," IEEE Trans. Automat. Contr., vol. AC-21, pp. 701-705, Oct. 1976.

[2] K. R. Krishnan and S. Brzezowski, "Design of robust linear regulator with prescribed trajectory insensitive to paramater variations," IEEE Trans. Automat. Contr., vol. AC-23, no. 3, pp. 474-478, June 1978.

Generic Pole Assignment Using Dynamic Output Feedback

Theodore E. Djaferis*

Abstract. A method is presented for assigning min(n+q, (q+1) m+q) closed loop poles of an nth order linear time invariant system (m outputs, ℓ inputs, m≥ℓ) using a linear, time invariant, proper, output feedback compensator of order q. It is also shown how the locations of remaining unassigned poles could be controlled. The compensator parameters are obtained by solving a linear matrix equation.

1. Introduction.

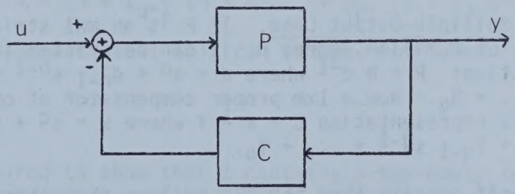
The problem of assigning the closed loop poles of a linear time invariant multivariable system using a proper, linear, time invariant, output feedback compensator continutes to be of great interest. Even though several outstanding issues remain, good progress has been made as evidenced by the interesting work of several researchers. Some recent work can be found in the references.

In particular Kimura [11], Davison and Wang [6] show that for a controllable observable plant (order n, m outputs, ℓ inputs) it is "almost always" possible to assign min(n,m+ℓ-1) closed loop poles arbitrarily close to a given set of real and complex conjugate values, by using constant output feedback. The issue of what happens to the remaining unassigned poles is not addressed. In a recent paper Antsaklis and Wolovich [1] present a different way of assigning min(n,ℓ+m-1) poles and suggest ways of dealing with the remaining unassigned poles. They also extend their result to include dynamic output feedback and show that if the original system is initially augmented by q integrators and then constant output feedback applied, m+ℓ+2q-1 poles can be assigned.

The present work deals with the question of how many poles can be asigned when the order of the compensator is fixed. It is shown that if a compensator of order q is used then min(n+q,(q+1)m+q) poles can be arbitrarily assigned. Throughout the paper it is assumed that m≥ℓ. This is not a restrictive assumption because the ℓ>m case can be treated in a very similar way and "dual" results obtained, (i.e. min(n+q,(q+1)ℓ+q poles can be assigned). The method of attack is different than the previous ones and it offers an improvement, in the appropriate cases, over the earlier result. It is further suggested how the locations of the remaining unassigned poles can be controlled.

2. Formulation.

The following feedback configuration is considered:



---

*The author is with the Department of Electrical and Computer Engineering at the University of Massachusetts, Amherst, Massachusetts 01003. This research has been supported by NSF under grant ECS-8006896 and partially supported by AFOSR under grant AFOSR-80-0155.

where P is a strictly proper $m x \ell$ input-output transfer function which represents the plant and C some $\ell x m$ proper dynamic compensator. Both P and C have elements in R(s) the field of rational functions in s over the reals R. The closed loop transfer function G is given by $G = P(I + CP)^{-1}$.

Using matrix fraction representations [7,10], $P = B_{RP}A_{RP}^{-1}$ some right representation of P, $N_{RP}D_{RP}^{-1}$ some right coprime representation of P, then G can be expressed as

$$G = B_{RP}(A_{LC}A_{RP} + B_{LC}B_{RP})^{1}A_{LC}$$

$$= D_{RP}(D_{LC}D_{RP} + N_{LC}N_{RP})^{-1}D_{LC} = N_{RP}\ \Phi^{-1}D_{LC}$$

$$= \tilde{N}_{RP}\ \tilde{\Phi}^{-1}\ \tilde{D}_{LC} \quad \text{(least order)},$$

where N_{RP}, $\tilde{\Phi}$ are right coprime, D_{LC}, $\tilde{\Phi}$ left coprime. It can be shown [5] that if $\phi(s)$ is the characteristic polynomial of the closed loop system then $\phi(s) = \alpha \det \Phi$ where α is some non-zero constant.

Now if $\phi(s) = s^i + \phi_{i-1}\ s^{i-1} + \phi_{i-2}\ s^{i-2} + \dots + \phi_0$ we know that $\underline{s} = (s_1,s_2,\dots,s_k)$ $k \le i$ are roots of $\phi(s)$ iff

$$[\phi_{i-1},\phi_{i-2},\dots,\phi_0]\ \underbrace{\begin{bmatrix} s_1^{i-1} & s_2^{i-1} & & s_k^{i-1} \\ s_1^{i-2} & s_1^{i-2} & & s_k^{i-2} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ s_1 & s_2 & & s_k \\ 1 & 1 & & 1 \end{bmatrix}}_{Q} = -[s_1^i,s_2^i,\dots,s_k^i]$$

where Q is an $i x k$ matrix.

Several definitions of genericity have been used. Throughout this paper a set $S \subset R^t$ will be called generic if it contains a non-empty Zariski open set of R^t [16].

3. Single-Input, Multiple-Output Case. If P is an $m x 1$ strictly proper transfer function, of McMillan degree n, it can be written in the right coprime representation: $P = N\ d^{-1}$ where $d = s^n + d_{n-1}\ s^{n-1} + \dots + d_0$, $N = N_{n-1}\ s^{n-1} + \dots + N_0$. Now a $1 x m$ proper compensator of order q can be written in the left representation $C = x^{-1}\ Y$ where $x = s^q + x_{q-1}\ s^{q-1} + \dots x_0$, $Y = Y_q\ s^q + Y_{q-1}\ s^{q-1} + \dots + Y_0$.

If x, Y are left coprime then the closed loop characteristic polynomial is: $\phi(s) = xd + YN = s^{n+q} + \phi_{n+q-1}\ s^{n+q-1} + \dots + \phi_0$. This relationship can be expressed in the following way:

$$[1, Y_q, x_{q-1}, Y_{q-1}, \dots, x_0, Y_0]\ S_{q+1}(d,N) = [1,\phi_{n+q-1}, \dots, \phi_0] \qquad (3.1)$$

where $S_{q+1}(d,N)$ is the $q+1^{th}$ order Sylvester Resultant [2] of d,N (a $(q+1)(mx1) \times (n+q+1)$ matrix).

From the above one can clearly see that the coefficients of the characteristic polynomial are linear functions of the compensator parameters. This consideration will play a crucial role in the $m \times \ell$ case where a compensator structure will be employed, that satisfies this condition.

Main Lemma

Let $P = Nd^{-1}$ be an mx1 strictly proper transfer function, with d of degree n, with $m|n$ and $q \geq 0$ a fixed integer. Let $k = (q+1)m + q \leq n+q = i$, and $\phi(s)$ be the closed loop characteristic polynomial and define:

$$W = \{(N,d) \in R^{mn+n} \mid d = s^n + d_{n-1}d^{n-1} + \dots + do, N = N_{n-1}s^{n-1} + \dots + N_0\}$$

$$S = \{(s_1 s_2, \dots, s_k) \in R^k \mid s_i \text{ real}\}$$

$$Z = \left\{(N,d,\underline{s}) \in R^{mn+n} \times R^k \;\middle|\; \begin{array}{l}\text{For which there exists a proper compensator} \\ \text{of order } q \text{ such that } s_1,s_2,\dots,s_k \text{ are roots} \\ \text{of } \phi(s)\end{array}\right\}$$

Then Z is a generic subset of $R^{mn+n} \times R^k$.

Remark: The requirement that $m|n$ is not restrictive and it is introduced for convenience. The general case can be treated with very minor alterations.

Let $\overline{S}_{q+1}$, be the submatrix obtained from $S_{q+1}(d,N)$ by removing the first row and first column and α the first row of $S_{q+1}(d,N)$ with the first entry removed.

Since $s_1,s_2,\dots,s_k$ will be the roots of $\phi(s)$ iff

$$[\phi_{i-1},\phi_{i-2},\dots,\phi_0]\, Q = -\,[s_1{}^i,s_2{}^i,\dots,s_k{}^i]$$

$s_1,s_2,\dots s_k$ will be the roots of $\phi(s)$ if we can find a $\underline{y}$

$$\underline{y} = [Y_q,\ x_{q-1},\ Y_{q-1},\dots,\ x_0,\ Y_0]$$

such that $\underline{y} \cdot \overline{S}_{q+1} \cdot Q = -\,[s_1{}^i,s_2{}^i,\dots,s_k{}^i] - \alpha \cdot Q.$ (3.2)

The compensator which will accomplish this is

$$C = x^{-1}\, Y, \quad x = s^q + x_{q-1}\, s^{q-1} + \dots + x_0, \quad Y = Y_q\, s^q + \dots + Y_0.$$

The Main Lemma suggests that this can be done for "almost all" N, d and $\underline{s} = (s_1,\dots,s_k)$.

Proof:

It is required to show that Z contains a non-empty Zariski open set. The matrix $\overline{S}_{q+1}\, Q$ is kxk. Let F $R^{mn+n} \times R^k$ for which it is invertible. $\overline{S}_{q+1}$ is rank k on a Zariski open set, which is non-empty since any Nd^{-1} with equal observability indices will belong to this set (Bitmead 1978). The matrix Q is also rank k for a generic subset of R^k since Q is rank k for any $\underline{s}$ for which the s_i are distinct. The product will be invertible for a generic subset of $R^{mn+n} \times R^k$. Equation (3.2) therefore does have a solution for a generic subset of $R^{mn+n} \times R^k$.

We further need to show that the x(s), Y(s) so constructed are generically left coprime. Clearly x(s), Y(s) are left coprime in a Zariski open set, we just need to show that this is non-empty. One can show that an $(N,d,\underline{s})$ which can be thought of as a point in $R^{mn+n} \times R^k$ space, can be found for which x(s), Y(s) are left coprime. This completes the proof.

The Main Lemma suggests that for "almost all" mx1 transfer functions of McMillan degree n and for "almost all" $\underline{s}, k = (q+1)m + q$ there exists a proper compensator of order q such that $s_1, s_2, \ldots, s_k$ are k roots of the closed loop characteristic polynomial $\phi(s)$. Since $\phi(s)$ is of degree n+q this means that in general there are n-(q+1)m unassigned poles. This means that some of these could even be unstable. As this is a matter of concern it is presently under investigation. It is possible that by restricting the assignable roots to lie in a certain region to assure that all of them are stable. In the general $m \times \ell$ case it is shown how additional compensator parameters can be introduced to help control the remaining unassigned poles.

One can show that if complex conjugate pairs are included in $\underline{s}$ the unique solution of (2.3), whenever it exists, is real.

Remark: It is interesting to note how the number of assignable poles increases as a function of q, the order of the compensator. Assuming that $m|n$ and since "generically" the observability indices of the plant are all equal to $\mu = n/m$, we see the following:

If q=0, m poles are arbitrarily assigned,

q=1, 2m+1 poles are arbitrarily assigned,

$\vdots$

$q=\mu-1$, $\mu m + \mu - 1 = n + \mu - 1$ poles are arbitrarily assigned,

which means that all closed loop pooles can be arbitrarily assigned. A similar result when $q=\mu-1$ has been obtained earlier by Brasch and Pearson [3] using a different approach.

An algorithm for constructing the solution involves expressing P in a right coprime representation and solving a linear equation over the reals (or complexes) since $\underline{y}$ from (3.2)

$$\underline{y} = - [s_1^i, s_2^i, \ldots, s_k^i](\bar{S}_{q+1} \cdot Q)^{-1} - \alpha Q(\bar{S}_{q+1} Q)^{-1}.$$

Comparing this procedure with the one suggested by Antsaklis and Wolovich 1970 for the mx1 case one can see that for a compensator of order q, using the present method (m+1) q + m poles are assigned, whereas with the earlier one 2q + m are assigned. It should also be pointed out that the method used there is different than the present one in this respect as well, in that initially P is augmented by q integrators and then constant output feedback is used to close the loop.

4. Multiple-Input, Multiple-Output Case. If P is an $m \times \ell$ $(m \geq \ell)$ strictly proper transfer function of McMillan degree n and equal controllability indices λ, (i.e. $n=\ell\lambda$) it can be expressed in the right coprime representation: $P = N_{RP} D_{RP}^{-1}$ where

$$D_{RP} = Is + D_{\lambda-1} s^{-1} + \ldots + D_o \qquad N_{RP} = \begin{bmatrix} N_{\lambda-1} \\ K_{\lambda-1} \end{bmatrix} s^{\lambda-1} + \ldots + \begin{bmatrix} N_o \\ K_o \end{bmatrix}. \qquad (4.1)$$

D_i, N_i are $\ell x \ell$ matrices, K_i are $(m-\ell)x\ell$ matrices. Now if an ℓxm proper compensator $C = X^{-1}Y$ is used of the form

$$X = X_q s^q + X_{q-1}s^{q-1} + \ldots + X_o \qquad Y = Y_q s^q + Y_{q-1}s^{q-1} + \ldots + Y_o$$

(X_i are $\ell x\ell$, Y_i are ℓxm), and X, Y are left coprime then the closed loop characteristic polynomial is given by $\phi(s) = \det(XD_{RP} + YN_{RP}) = \det\Phi$

If one uses Sylvester Resultants [2] to express the relationships XD_{RP} + XN_{RP} one has the following

$$[X_q, Y_q, \ldots, X_o, Y_o] \begin{bmatrix} I, & D_{\lambda-1}, & D_{\lambda-2}, & \ldots, & D_o, & 0, & \ldots, & 0 \\ 0 & N_{\lambda-1}, & N_{\lambda-2}, & & N_o, & 0, & \ldots, & 0 \\ & & & \cdots & & & & \\ 0 & K_{\lambda-1}, & K_{\lambda-2}, & & K_o, & 0, & \ldots, & 0 \\ & & & & \vdots & & & \\ & & & I & & D_{\lambda-1}, & D_{\lambda-2}, \ldots, & 0 \\ & & & 0 & & N_{\lambda-1}, & N_{\lambda-2}, \ldots, & 0 \\ & & & 0 & & K_{\lambda-1}, & K_{\lambda-2}, \ldots, & 0 \end{bmatrix} \begin{matrix} \\ \\ \text{q+1 block} \\ \text{rows} \\ \\ \\ \end{matrix} \qquad (4.2)$$

$$= [\Phi_{\lambda+q}, \Phi_{\lambda+q-1}, \ldots, \Phi_o] \,,$$

where $\quad \Phi = \Phi_{\lambda+q}s^{\lambda+q} + \ldots + \Phi_o$, is $\ell x\ell$.

Now since $\phi(s) = \det \Phi$, it is evident that in general the coefficients of $\phi(s)$ are non-linear expressions in the compensator parameters. This is precisely where many difficulties concerning the pole assignment problem lie. One way of proceedig is to find ways of exploiting this non-linear structure. In this paper a successful approach is presented which "avoids" the non-linear analysis. The problem is formulated in such a way that the nonlinear structure is forced to become linear.

Suppose that the compensators under consideration are restricted to have the following structure.

$$X = \underbrace{\begin{bmatrix} 1,0,0,0 \ldots 0 \\ 0 \\ \vdots \quad 0_{\ell-1} \\ 0 \end{bmatrix}}_{X_q} s^q + \underbrace{\begin{bmatrix} {}_{q-1}X_1, 0 \ldots 0 \\ 0 \\ \vdots \quad 0_{\ell-1} \\ 0 \end{bmatrix}}_{X_{q-1}} s^{q-1} + \ldots + \underbrace{\begin{bmatrix} {}_0X_1, 0 \ldots 0 \\ 0 \\ \vdots \quad I_{\ell-1} \\ 0 \end{bmatrix}}_{X_0}$$

$$Y = \underbrace{\begin{bmatrix} {}_q y_1, {}_1 y_2, \cdots {}_q y_\ell \cdots {}_q y_m \\ \vdots \quad 0_{\ell-1} \qquad 0 \\ 0 \end{bmatrix}}_{Y_q} s^q + \ldots + \underbrace{\begin{bmatrix} {}_0 y_1, {}_0 y_2, \cdots {}_0 y_\ell \cdots {}_0 y_m \\ \vdots \quad I_{\ell-1} \qquad 0 \\ 0 \end{bmatrix}}_{Y_0} .$$

By construction $C = X^{-1}Y$ is proper. Now if $\underline{y}(s)$ is the first row of Y, $(d_{11}(s), d_{12}(s), \ldots, d_{1\ell}(s))$ the first row of $\bar{D}_{RP}$, $\underline{n}_j(s)$ the j^{th} column of N_{RP} and $(\phi_{11}(s), \phi_{12}(s), \ldots, \phi_{1\ell}(s))$ the first row of Φ one has the following:

$$x(s)d_{11}(s) + \underline{y}(s)\underline{n}_1(s) = \phi_{11}(s)$$
$$x(s)d_{12}(s) + \underline{y}(s)\underline{n}_2(s) = \phi_{12}(s)$$
$$\vdots$$
$$x(s)d_{1\ell}(s) + \underline{y}(s)\underline{n}_\ell(s) = \phi_{1\ell}(s) .$$

Let $\bar{\Phi}$ be the $(\ell-1)\times\ell$ matrix which contains the remaining $\ell-1$ rows of Φ. Then $\bar{\Phi} = [0, I_{\lambda-1}]s^\lambda + (\bar{D}_{\lambda-1}+\bar{N}_{\lambda-1})s^{\lambda-1} + \ldots + (\bar{D}_0+\bar{N}_0)$ where $\bar{D}_i$, $\bar{N}_i$ are obtained from D_i, N_i by removing the first row.

Now since the closed loop characteristic polynomial is $\phi(s) = \det\Phi$ one can easily see by expanding by the first row that:

$$\phi(s) = \phi_{11}(s)\Delta_{11} + \phi_{12}(s)\Delta_{12} + \ldots + \phi_{1\ell}(s)\Delta_{1\ell} \tag{4.4}$$

(Δ_{1i} is the appropriate $(\ell-1)\times(\ell-1)$ minor of Φ). The coefficients of $\phi(s)$ are linear expressions in the compensator parameters. This relationship is made more explicit in the following:

$$\phi(s) = x(s)(\underbrace{d_{11}(s)\Delta_{11} + \ldots + d_{1\ell}(s)\Delta_{1\ell}}_{d}) + \underline{y}(s)(\underbrace{\underline{n}_1(s)\Delta_{11} + \ldots + \underline{n}_\ell(s)\Delta_{1\ell}}_{= N})$$
$$= x(s)d + \underline{y}(s)N. \tag{4.5}$$

Since d is a polynomial of degree $\lambda\ell = n$, N an $m\times1$ vector of degree $\lambda\ell-1 = n-1$, $x(s)$ a polynomial of degree q and $\underline{y}(s)$ a $1\times m$ vector of degree q this fits precisely the formulation used in the analysis of $m\times1$ systems. Namely (4.5) can be expressed as:

$$[1, Y_q, x_{q-1}, \ldots, x_0, Y_0]S_{q+1}(d,N) = [1, \phi_{\lambda+q}, \ldots \phi_0] \tag{4.6}$$

This would indicate that the Main Lemma could somehow be used in obtaining a result for the $m\times\ell$ case. In a sense the original $m\times\ell$ system has been "transformed" (or reduced) to the $m\times1$ system corresponding to (4.5). This "transformation" will play a crucial role in the proof of the upcoming Theorem, and is therefore now made more precise.

Let the given transfer function (4.1) be parameterized as follows:

$$D_{RP} = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix} s^{\lambda} + \underbrace{\begin{bmatrix} a_1, & \dots a_\ell \\ a_{\lambda\ell+1}, & \dots a_{(\lambda+1)\ell} \\ & \vdots \\ a_{(\lambda-1)\lambda\ell+1} \dots & \end{bmatrix}}_{D_{\lambda-1}} s^{\lambda-1} + \dots + \underbrace{\begin{bmatrix} a_{(\lambda-1)\ell+1} \dots a_{\lambda\ell} \\ a_{2\lambda\ell} \\ \\ s_{\ell.\lambda.\ell} \end{bmatrix}}_{D_0}$$

$$N_{RP} = \begin{array}{r} \ell\left\{ \begin{matrix} \\ \\ \\ \end{matrix} \right. \\ m-\ell\left\{ \begin{matrix} \\ \\ \\ \end{matrix} \right. \end{array} \underbrace{\begin{bmatrix} a_{\lambda\ell^2+1} & \dots & a_{\lambda\ell^2+\ell} \\ & \vdots & \\ a_{\lambda\ell^2+(\lambda-1)\lambda\ell+1} & \dots & \\ a_{2\lambda\ell^2+1} & \dots & \\ & \vdots & \\ a_{2\lambda\ell^2+(m-\ell-1)\lambda\ell+1} & \dots & \end{bmatrix}}_{N_{\lambda-1}} s^{\lambda-1} + \dots + \left.\underbrace{\begin{bmatrix} a_{\lambda\ell^2+(\lambda-1)\ell+1} & \dots & a_{\lambda\ell^2+\lambda\ell} \\ & \vdots & \\ \dots & a_{\lambda\ell^2+\lambda\ell^2} & \\ \dots & a_{2\lambda\ell^2+\lambda\ell} & \\ & \vdots & \\ \dots & a_{2\lambda\ell^2+(m-\ell)\lambda\ell} & \end{bmatrix}}_{N_0}\right\} \ell$$

Let the reduced transfer function obtained in (4.5) be parameterized as follows:

$$d = s^n + b_1 s^{n-1} + b_2 s^{n-2} + \dots + b_n$$

$$N = \begin{bmatrix} b_{n+1} \\ b_{2n+1} \\ \vdots \\ b_{\ell n+1} \\ b_{(\ell+1)n+1} \\ \vdots \end{bmatrix} s^{n-1} + \begin{bmatrix} b_{n+2} \\ b_{2n+2} \\ \vdots \\ b_{\ell n+2} \\ b_{(\ell+1)n+2} \\ \vdots \end{bmatrix} s^{n-2} + \dots + \begin{bmatrix} b_{n+n} \\ b_{3n} \\ \vdots \\ b_{(\ell+1)n} \\ b_{(\ell+2)n} \\ \vdots \\ b_{(m+1)n} \end{bmatrix} \Big\} \ell$$

Let $\underline{a} = (a_1, a_2, \dots, a_{\lambda\ell(m+\ell)})$. It is clear from (4.5) that each b_j is a function of $\underline{a}$ and will be indicated as such $b_j(\underline{a})$ whenever necessary. It is in fact a polynomial in $\underline{a}$.

Let $f\colon\ R^{n(m+\ell)} \to R^{n(m+1)}$ be defined as:

$$f(\underline{a}) = (b_1, b_2, \dots, b_{(m+1)n}) = \underline{b}$$

The function f therefore describes the "transformation" precisely. The structure of f will play a very important role in the proof of the main result where it will be required to show that the Jacobian of f, J_f is full rank at some point $\underline{a}$. The following Proposition addresses this issue.

Proposition. Let J_f be the Jacobian of f. There exists a point $\underline{a}$ (i.e. a specific transfer function of the type (4.1)) such that $J_f(\underline{a})$ is full rank.

The main result of the paper can now be stated and proved.

Theorem

Let $P = N_{RP}D_{RP}^{-1}$ be an $m \times \ell$ $(m \geq \ell)$ strictly proper transfer function where

$$D_{RP} = Is^{\lambda} + D_{\lambda-1}s^{\lambda-1} + \ldots + D_0, \quad N_{RP} = \bar{\bar{N}}_{\lambda-1}s^{\lambda-1} + \ldots + \bar{\bar{N}}_0 .$$

Let $m|n$ and $q \geq 0$ a fixed integer. Let $k = (q+1)m + q \leq n + q = i$, $\phi(s)$ the closed loop characteristic polynomial and define:

$$W = \{(N_{RP}, D_{RP}) \varepsilon R^{(m+\ell)n} \,|\, D_{RP} = Is^{\lambda} + D_{\lambda-1}s^{\lambda-1} + \ldots + D_0,\ N_{RP} = \bar{\bar{N}}_{\lambda-1}s^{\lambda-1} + \ldots + \bar{\bar{N}}_0\}$$

$$S = \{(s_1, s_2, \ldots, s_k) \ \varepsilon\ R^k \,|\, s_i \text{ real }\}$$

$$Z = \{(N_{RP}, D_{RP}, \underline{s}) \varepsilon R^{(m+\ell)n} \times R^k \ \left|\ \begin{array}{l}\text{For which there exists a proper}\\ \text{compensator of order } q \text{ such that}\\ s_1, s_2, \ldots, s_k \text{ are roots of } \phi(s)\end{array}\right.\}$$

Then Z is a generic subset of $R^{(m+\ell)n} \times R^k$.

Remark: The requirement that $m|n$ is introduced merely for convenience.

Proof:

From (4.6) let $\underline{y} = [Y_q, x_{q-1}, \ldots, x_0, Y_0]$. Then as in the mx1 case

$$\underline{y} \cdot \bar{S}_{q+1}(d,N)Q = -[s_1^{\,i}, s_2^{\,i}, \ldots, s_k^{\,i}] - \alpha \cdot Q. \tag{4.7}$$

This follows directly from the reformulation of the $m \times \ell$ problem as an mx1 problem. $\bar{S}_{q+1}(d,N)$ is a matrix whose entries are polynomials in $\underline{a}$. Now the set $E \subset R^{(m \times \ell)n} \times R^k$ for which a solution to (4.7) exists and is such that the corresponding $x(s)$, $\underline{y}(s)$ are left coprime is a Zariski open set. For it to be generic it must be shown to be non-empty. The Main Lemma guarantees this to be true for almost all $\underline{b} \times \underline{s}$. Since there exists an $\underline{a}$ (by the Proposition) for which $J_f(\underline{a})$ is full rank, then there exists an open set U such that $f(\underline{a}) \varepsilon U$ and $\bar{f}(R^{(m \times \ell)n}) \supset U$, (by the inverse function theorem [12]). This means that E contains at least one point. This completes the proof of the Theorem.

The Theorem suggests that for "almost all" $m \times \ell$ transfer functions of McMillan degree n (and equal controllability ilndices $\lambda = n/\ell$) and for "almost all" $\underline{s}$, $(k = (q+1)m + q)$ there exists a proper compensator of order q such that $s_1, s_2, \ldots, s_k$ are k roots of the closed loop characteristic polynomial $\phi(s)$. As in the mx1 case some roots may be left unassigned. In the multi-input case the possibility does exist for introducing additional parameters in the compensator that can be used to control the remaining roots. One such possibility is to modify the original compensator structure. Let $C = X^{-1}Y$,

$$X = \begin{bmatrix} x(s), & 0, & \dots, & 0 \\ 0 & & & \\ \vdots & & & \\ & & I_{\ell-1} & \\ 0 & & & \end{bmatrix} \quad Y = \begin{bmatrix} y_1(s), y_2(s), \dots, & y_\ell(s), \dots, & y_m(s) \\ 0 & & \\ \vdots & L & 0 \\ 0 & & \end{bmatrix} \qquad (4.8)$$

where L is an $\ell-1 \times \ell-1$ constant matrix containing $(\ell-1)^2$ free parameters comprising a vector $\underline{c}$. The compensator $C = X^{-1}Y$ is still proper of order q. Following a similar proof to the one given one can show that k poles can be arbitrarily assigned for almost all $\underline{c} \times \underline{a} \times \underline{s} \in R^{(\ell-1)^2} \times R^{(m\times\ell)n} \times R^k$. This means that for "almost all" choices of $\underline{a}$ and $\underline{s}$ a proper compensator of the form given above, which is parameterized by $\underline{c}$, (valid for "almost all" $\underline{c}$), can be constructed which assigns k poles of the closed loop system to $\underline{s} = (s_1, s_2, \dots, s_k)$. The freedom afforded by the presence of these parameters can then be used to "control" the location of the remaining unassigned poles. An illustrative example is given below.

Remark: As in the mx1 case the number of assignable poles increases as a function of q in such a way that if $1 = \mu-1$ then all the closed loop poles are arbitrarily assigned. Brasch and Pearson [3] show, in an entirely different way, that for a controllable observable system adding of a $\mu-1$ order compensator is sufficient to ensure arbitrary pole assignment.

Example: Let $m=2$, $\ell=2$, $n=4$ and

$$D_{RP} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} s^2 \quad N_{RP} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} s + \begin{bmatrix} 0 & -1 \\ 1 & 1 \end{bmatrix} \quad X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} y_1 & y_2 \\ 0 & c \end{bmatrix},$$

using the modified compensator structure given in (4.8).

This transfer function is reduced to the 2x1 transfer function

$$d = s^4 + cs^2 \qquad N = \begin{bmatrix} 0 \\ 1 \end{bmatrix} s^3 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} s^2 + \begin{bmatrix} c \\ 0 \end{bmatrix} s + \begin{bmatrix} c \\ 0 \end{bmatrix}.$$

The results suggest that with a constant compensator (q=0), 2 poles can be assigned. Let $s_1 = (-1 + j2)$, $s_2 = (-1 - j2)$. Then from (3.2)

$$C = \begin{bmatrix} \frac{(c-3)25}{8c} & \frac{7+3c}{8} \\ 0 & c \end{bmatrix}.$$

This compensator makes the closed loop characteristic polynomial equal to

$$\phi(s) = (s^2 + 2s + 5)(s^2 + 3/8(c-3)s + 5/8(c-3)).$$

One can easily see that for "almost all" choices of c(i.e. $c \neq 0$) the compensator makes $-1 + j2$ and $-1 - j2$ two of the closed loop poles. If $c>3$, s_3 and s_4 are guaranteed to be stable.

The above suggestion becomes a very powerful tool in that the compensator C is given, parameterized by c, that assigns k of the closed loop poles. Since the remaining poles in general depend on c, they can in turn be controlled.

References

[1] Antsaklis P. J., Wolovich W. A., 1977, Int. J. Control, Vol. 25, No. 6.

[2] Bitmead R. R., Kung S. Y., Anderson B. D. O., Kailath T., 1978, IEEE Trans. on AC, Vol. AC-23, No. 6.

[3] Brasch F. M., Pearson J. B., 1970, IEEE Trans. on AC, Vol. AC-15, No. 1.

[4] Brockett R. W., Byrnes C. I., 1981, IEEE Trans. on AC, Vol. AC-26, No. 1.

[5] Chen C. T., Hsu C. H., 1968, Proc. IEEE, 56(11), 2061-2062.

[6] Davison E. J., Wang S. H., 1975, IEEE Trans. on AC, Vol. AC-20 No. 4.

[7] Desoer C. A., Vidyasagar M., 1975, Fedback Systems: Input-Output Properties (Academic Press, New York).

[8] Djaferis T. E., Mitter S. K., 1979, Proceedings of 1979 IEEE CDC, Fort Lauderdale, Florida.

[9] Emre E., 1980, Siam J. Control and Opt., Vol. 18, No. 6.

[10] Kailath T., 1980, Linear Systems, (Prentice Hall, Englewood Cliffs, NJ).

[11] Kimura H., 1975, IEEE Trans. on AC, Vol. AC-25, No. 4.

[12] Luenberger D. G., 1969 Optimization by Vector Space Methods, (John Wiley, NY).

[13] Rosenbrock H. H., Hayton G. E., 1978, Int. J. Control, Vol. 27, No. 6.

[14] Willems J. C., Hesselink W. H., 1978, Proceedings 1978 IFAC Congress, Helsinki, Finland.

[15] Wolovich W. A., 1974, Linear Multivariable Systems (Springer Verlag, New York).

[16] Zariski O. Samuel, 1958, Commutative Algebra Vol. 1, (Van Nostrand, Princeton, NJ).

CONTINUOUS-TIME STOCHASTIC MODEL SIMPLIFICATION

JITENDRA K. TUGNAIT
Dept. of Electrical and Computer Engineering
The University of Iowa, Iowa City, Iowa 52242

ABSTRACT

Approximation of high-order and time varying linear continuous-time Gaussian models by low-order and time-invariant models is considered. The recent work of Baram and Be'eri pertaining to the discrete-time systems is extended to continuous-time system models. The model simplification is carried out by maximizing the probabilistic ambiguity between the actual system and the approximate model. Differences between the continuous-time and the discrete-time model simplification problems are examined. The case when the observation noise covariance of the approximate model differs from that of the actual system, presents technical difficulties not present in the discrete-time version. Numerical examples are presented to illustrate the proposed approach.

I. INTRODUCTION

This paper is concerned with the problem of approximating high-order, time-varying linear continuous-time stochastic models by low-order, time-invariant models. The main objective is to extend the recent work of Baram and Be'eri [1], pertaining to the discrete-time models, to continuous-time models. As in [1], we employ state-space models in the time-domain (driven by and observed in white Gaussian noise), which makes it possible to take into account stochastic disturbances, deterministic inputs, and initial conditions.

For a background to the problem, we refer the reader to [1]. To the list of references in [1], we add [2], [3], and [4].

Following [1], the model simplification is carried out by maximizing the probabilistic ambiguity between the actual system and the simplified model. An approximate coefficient, which is different from the one in [1], is defined. We deal only with the structure reduction problem. The set reduction problem can be handled just as in [1]. Unlike the discrete-time case, the probability density of continuous-time observations (on a finite-time interval), indexed by a given simplified model, is ill-defined; therefore, the approximation criterion used in this paper is different from that in [1], however, it is in the same spirit.

II. STRUCTURE (ORDER AND TIME) REDUCTION

A. General

Consider continuous-time observations $\lambda_T \triangleq \{y(t), t\in[0,T]\}$ over a finite time interval $[0,T]$ with $y(t)\in R^m$. Let $\bar{M} = \{M(\theta), \theta\in\Theta\}$ denote a set of (simplified) models for λ_T, depending upon a parameter vector θ. To each $\theta\in\Theta$ there corresponds a probability measure $p_\theta(\lambda_T)$ induced by $y(t)$ on ₵ $= C^m[0,T]$ where ₵ denotes the space of continuous functions from $[0,T]$ into R^m. Assume that $p_\theta(\lambda_T)$ and $p_0(\lambda_T)$ are mutually absolutely continuous where $p_0(\lambda_T)$ denotes the known (or actual) probability measure of λ_T. Following [1] (see also [2] and [4]), an approximation criterion for selecting a model for λ_T from $\bar{M}$ is the one that maximizes the Kullback-Leibler information between the true and the approximating probability measures, i.e.,

$$\max_{\theta\epsilon\Theta} E_0 \left\{ \ln \frac{dp_\theta(\lambda_T)}{dp_0(\lambda_T)} \right\} \tag{1}$$

where the subscript 0 on the expectation operator E in (1) denotes that it is with respect to (w.r.t.) the known distribution of $\lambda_T, p_0(\lambda_T)$, and $dp_\theta(\lambda_T)/dp_0(\lambda_T)$ denotes the Radon-Nikodym (R-N) derivative of p_θ w.r.t. p_0.

Let $p_W(\lambda_T)$ denote a (fixed) probability measure on ¢, not dependent upon $\theta\epsilon\Theta$ or on the actual system model, such that $p_\theta(\lambda_T)$ and $p_0(\lambda_T)$ are mutually absolutely continuous w.r.t. $p_W(\lambda_T)$ for each $\theta\epsilon\Theta$. Such a measure can always be found for Gauss-Markov models with noise corrupted observations where the observation noise covariance is known (fixed) and is identical for the entire model set $\bar{M}$ and the actual system; see Section IIB. In this case, it is convenient to define the approximation criterion as

$$\max_{\theta\epsilon\Theta} E_0 \left\{ \ln \frac{dp_\theta(\lambda_T)}{dp_W(\lambda_T)} \right\} \tag{2}$$

Note that the criteria (1) and (2) coincide.

Following [1], and without loss of generality, the approximation criterion (2) may be written as

$$\min_{\theta\epsilon\Theta} J(\theta) \tag{3}$$

where

$$J(\theta) = -\frac{2}{T} E_0 \left\{ \ln \frac{dp_\theta(\lambda_T)}{dp_W(\lambda_T)} \right\} \tag{4}$$

Now define

$$K' = (J(*) - J(\theta))/J(*) \tag{5}$$

where

$$J(*) = -\frac{2}{T} E_0 \left\{ \ln \frac{dp_0(\lambda_T)}{dp_W(\lambda_T)} \right\} \tag{6}$$

Now we wish to show that $K' \geq 0$. To this end, let

$$\exp(\nu_T) \triangleq \frac{dp_\theta(\lambda_T)}{dp_0(\lambda_T)} \tag{7}$$

From Jensen's inequality [5], we have

$$E_0\{\nu_T\} \leq \ln E_0\{\exp(\nu_T)\} = 0 \tag{8}$$

Furthermore, we have

$$\frac{dp_\theta(\lambda_T)}{dp_0(\lambda_T)} = \frac{dp_\theta(\lambda_T)}{dp_W(\lambda_T)} / \left[\frac{dp_0(\lambda_T)}{dp_W(\lambda_T)} \right] \tag{9}$$

so that

$$E_0\{-\frac{2}{T} \nu_T\} = J(\theta) - J(*) \tag{10}$$

which together with (8) implies that $J(\theta)-J(*) \geq 0$.

Similarly, it is easy to show that

$$E_0 \left\{ \ell n \frac{dp_W(\lambda_T)}{dp_0(\lambda_T)} \right\} \leq 0$$

so that $J(*) \leq 0$. [Note that if p_1 and p_2 are two mutually absolutely continuous probability measures then $\frac{dp_1}{dp_2} = \left[\frac{dp_2}{dp_1}\right]^{-1}$ a.e.]. Hence, $K' \geq 0$. Notice that $J(\theta)$ can be nonnegative. However, for all "reasonable" approximating models we must have $J(\theta) < 0$. For, if $J(\theta) > 0$, then a better approximating model (in the sense of (2)) would be to find a parameter vector θ' (not necessarily in Θ) such that $p_{\theta'}(\lambda_T) \equiv p_W(\lambda_T)$ so that $J(\theta') = 0$. For Gauss-Markov models, we show in Section IIB that $p_W(\lambda_T)$ may be chosen to be a Wiener measure with covariance equal to the observation noise covariance. Therefore, a "reasonable" approximating model should provide a better model for given λ_T than is provided by modelling λ_T as a Wiener process (or white Gaussian noise). Assuming that the approximating model set $\bar{M}$ has been selected to have $J(\theta) \leq 0 \; \forall \theta \epsilon \Theta$, it follows that

$$0 \leq K' \leq 1. \tag{11}$$

It should be noted that K' given by (5) plays the same role as the K' of [1], although the two are defined differently. The quantity K' measures the difference between the approximate model and the actual one. Following [1], a measure of approximation is defined as the complement of K', i.e.,

$$K \triangleq 1-K' = \frac{J(\theta)}{J(*)} \tag{12}$$

It follows, therefore, that

$$0 \leq K \leq 1 \tag{13}$$

As in [1], we shall call K the approximation coefficient.

Remark 1. If $\bar{M}$ is not restricted to have $J(\theta) \leq 0$ for each $\theta \epsilon \Theta$, then we may have $K' > 1$ although still $K' \geq 0$. This leads to possible negative values of K although $K \leq 1$. Note that higher values of K indicates better approximation (in the sense of (3)). Negative value of K implies that the approximate model under consideration is inferior to the white noise model for λ_T in the Gauss-Markov case.

B. Linear Gaussian Systems

Consider the following continuous-time linear model in state-space form:

$$dx(t) = A(t)x(t)dt + B(t)u(t)dt + F(t)dW_1(t) \tag{14}$$

$$dy(t) = C(t)x(t)dt + G(t)dW_2(t), \; t \epsilon [0,T] \tag{15}$$

where u(t) is a p-dimensional known input function; x(t) and y(t) are n- and m-dimensional state and observation vectors, respectively; x(0) is normally distributed with mean x_0 and covariance P(0); A(t), B(t), F(t), C(t) and G(t) are real time-continuous known matrices of appropriate dimensions; and $W_1(t)$ and $W_2(t)$ are n- and m-dimensional independent

Wiener processes, respectively, with identity covariances. We take $W_1(0) = W_2(0) = 0$. Finally, we assume that $G(t)G^*(t) > 0$ for each $t \in [0,T]$ to exclude the singular case. (G^* denotes transpose of G and $GG^* > 0$ denotes that the matrix GG^* is positive definite.) The given model is denoted by

$$M = \{(A(t),B(t),F(t),C(t),G(t),x(0),P(0)), t \in [0,T]\} \tag{16}$$

A time-invariant model of the form

$$dx(t) = Ax(t)dt + Bu(t)dt + FdW_1(t) \tag{17}$$

$$dy(t) = Cx(t)dt + GdW_2(t), \ t \in [0,T] \tag{18}$$

that approximates M in some sense is required. This simplified model may be of order less than n. The simplified model set is denoted by

$$\bar{M} = \{M_\theta = (A_\theta,B_\theta,F_\theta,C_\theta,G_\theta,x_\theta(0),P_\theta(0)); \ \theta \in \Theta\} \tag{19}$$

where θ is the vector of free parameters of the simplified model and Θ is the parameter set.

In the rest of the paper we shall take for each $\theta \in \Theta, G(t) \equiv G_\theta \equiv G$, a completely fixed constant matrix, i.e., G has no free parameters and it is also not a function of time. The reasons for this restriction are technical and may be found in [10]. Note that no such difficulty arises in the discrete-time case. In order to calculate the approximation criterion for the given problem first we have to choose a suitable probability measure for $p_W(\lambda_T)$ introduced in Section IIA. We take it to be $p_{GW_2}(\lambda_T)$, the probability measure induced by the Wiener process $GW_2(t)$ on ₵. Then from [6, Chapter 8] and the Feldman-Hajek dichotomy theorem [8, Chapter 6], it follows that $p_0(\lambda_T)$, $p_{GW_2}(\lambda_T)$ and $p_\theta(\lambda_T)$ for each $\theta \in \Theta$, are all mutually absolutely continuous w.r.t. each other. [Note that $p_{G_1W_2}(\lambda_T)$ and $p_{G_2W_2}(\lambda_T)$ are singular if $G_1G_1^* \neq G_2G_2^*$ where $W_2(t)$ is a Wiener process; this is why we have taken $G(t) \equiv G_\theta \equiv G$ in this section. See also [9] for related discussion.] Then we have [4], [7] (see Appendix for details).

$$J(\theta) = \frac{1}{T} \left[\int_0^T \operatorname{tr}(GG^*)^{-1}\{P_1{}^0(t;\theta) - S(t)\}dt \right] \tag{20}$$

where

$$P_1^0(t;\theta) = E_0\{[C(t)x(t) - C_\theta\hat{x}(t;\theta)][C(t)x(t) - C_\theta\hat{x}(t;\theta)]^*\} \tag{21}$$

$$S(t) = E_0 \ \{C(t)x(t)x^*(t)C^*(t)\} \tag{22}$$

$$\hat{x}(t;\theta) = E_\theta\{x(t)|\lambda_t\} \tag{23}$$

$$\lambda_t = \{y(\tau), \tau \in [0,t]\} \tag{24}$$

and E_θ denotes that the expectation is taken assuming the simplified model indexed by θ is true. Note that $S(t)$ is not a function of θ.

The first and the second derivatives of $J(\theta)$ w.r.t. θ, needed for a gradient-type minimization procedure, may be derived as in [1]. We omit the derivation; only the expressions for quantities needed to calculate $J(\theta)$ are given in the Appendix.

C. Time Invariant Systems: Steady State

Suppose that the original system is time-invariant and emphasis is on its steady-state characteristics. Then in the simplified model design, the approximation criterion takes its steady state value, i.e.,

$$\begin{aligned} J_\infty(\theta) &= \lim_{T\to\infty} J(\theta) \\ &= \mathrm{tr}(GG^*)^{-1}\{P_1{}^0(\theta)-S\} \end{aligned} \tag{25}$$

where $P_1{}^0(\theta)$ and S are the steady-state values of $P_1{}^0(t;\theta)$ and $S(t)$, respectively. Sufficient conditions for the existence of the steady-state values may be found in [4] (see also [7]). Finally, analyticity of $J_\infty(\theta)$ w.r.t. θ may be proved using the results in [11, Lemma 2.3]; this result is of theoretical significance for a gradient-type optimization of $J_\infty(\theta)$.

D. Example 1

We now consider an example to illustrate the usefulness of the proposed approach. Only the problem of steady-state order reduction for time invariant models is considered; other examples such as initial response approximation, or approximation of a time-varying system by a time-invariant model can also be devised following [1], but are not considered in this paper since the efficacy of the approach has already been established in [1].

Given a second-order system

$$dx(t) = \begin{bmatrix} 0 & 1 \\ -a & -1 \end{bmatrix} x(t)dt + \begin{bmatrix} 0 \\ 1 \end{bmatrix} dW_1(t) \tag{26}$$

$$dy(t) = [1 \quad 0] \;\; x(t)dt + dW_2(t), \tag{27}$$

it is desired to find a first-order model of the form

$$dx(t) = \theta x(t)dt + dW_1(t) \tag{28}$$

$$dy(t) = x(t)dt + dW_2(t) \tag{29}$$

$$\theta\epsilon\Theta = (-\infty,\infty)$$

that approximates the given model. We shall consider only a steady-state approximation. The optimal values (in the sense of (3)) of θ and the corresponding approximation coefficient K for several values of a are given in Table 1. The poles of the given system and the approximating models in the s-plane are shown in Fig. 1. The results shown in Table 1 and Fig. 1 are similar to those obtained in Example 1 of [1]. (In order to compare our results with those in Example 1 of [1] one must keep in mind the relationship between the s-plane and the z-plane poles of a continuous-time system and its discrete-time equivalent, respectively.)

When the original system has a dominant (low-frequency) pole (Case 1), the optimal simplified scalar model has its pole practically identical to the low-frequency pole. As the poles of the original system come closer (Cases 2-4) the pole of the approximate model lies to their right to yield similar frequency response to that of the original poles. When the poles of the original system are complex (Cases 5 and 6), the approximate model's pole moves to left as the imaginary part of the original system's poles increases, giving similar frequency response characteristics. This also leads to a lower approximation coefficient indicating that a steady-state approximation may not be quite satisfactory when the original system has oscillatory characteristics (complex poles); one may want to use a transient response approximation as discussed in [1].

III. CONCLUDING REMARKS

The problem of continuous-time stochastic model simplification has been treated. The approximation of high-order and time-varying systems by low-order and time-invariant models was achieved by maximizing the probabilistic ambiguity between the actual system and the approximate model. The main objective was to extend the recent work of Baram and Be'eri [1], pertaining to the discrete-time systems, to continuous-time system models. The points of similarities and of differences between the continuous-time and the discrete-time cases were examined. We considered only the structure reduction problem. The set reduction problem can be handled just as in [1].

The case when $G(t) \not\equiv G_\theta \not\equiv G$ for $\theta \in \Theta$, can be handled following the "trick" of [9] in defining a pseudo likelihood functional; see also [10]. Details are omitted due to space limitations.

REFERENCES

[1] Y. Baram and Y. Be'eri, "Stochastic model simplification," IEEE Trans. Automatic Control, vol. AC-26, pp. 379-390, April 1981.

[2] B.D.O. Anderson, J.B. Moore and R.M. Hawkes, "Model approximation via prediction error identification," Automatica, vol. 14, pp. 615-622, November 1978.

[3] J. Hickin and N.K. Sinha, "Model reduction for linear multivariable systems," IEEE Trans. Automatic Control, vol. AC-25, pp. 1121-1127, December 1980.

[4] J.K. Tugnait, "Identification and model approximation for continuous-time systems on finite parameter sets," IEEE Trans. Automatic Control, vol. AC-25, pp. 1202-1206, December 1980.

[5] R.B. Ash, Real Analysis and Probability. New York: Academic, 1972.

[6] A.V. Balakrishnan, Stochastic Differential Systems I, Lecture Notes in Economics and Mathematical Systems. Berlin: Springer-Verlag, 1973.

[7] J.K. Tugnait, "Convergence of continuous-time partitioned adaptive state estimators," Int. J. Control, vol. 33, No. 5, pp. 923-933, 1981.

[8] E. Wong, Stochastic Processes in Information and Dynamical Systems. New York: McGraw-Hill, 1971.

[9] A. Bagchi, "Continuous time systems identification with unknown noise covariance," Automatica, vol. 11, pp. 533-536, September 1975.

[10] J.K. Tugnait, "Global identification of continuous-time systems with unknown noise covariance," in Proc. 17th Annual Allerton Conf. Communication, Control, Computing, Univ. of Illinois, Urbana-Champaign, pp. 876-885, October 1979. (Also, to appear in IEEE Trans. Information Theory).

[11] C.I. Byrnes, "On the stabilizability of linear control systems depending on parameters," in Proc. 18th IEEE Conf. Decision and Control, Fort Lauderdale, Florida, pp. 233-236, December 1979.

APPENDIX

First, we derive (20). From [6, Theorem 8.1], we have

$$\frac{dp_\theta(\lambda_T)}{dp_{GW_2}(\lambda_T)} = \exp\{-\frac{T}{2}\, q[\theta;y(.);T]\} \tag{A1}$$

where

$$q[\theta;y(.);T] = \frac{1}{T}\,[\int_0^T \hat{x}^*(t;\theta)C_\theta^*(GG^*)^{-1}C_\theta\hat{x}(t;\theta)dt$$

$$-2\int_0^T \hat{x}^*(t;\theta)C_\theta^*(GG^*)^{-1}dy(t)]\ , \tag{A2}$$

$\hat{x}(t;\theta)$ is given by (23), and the second integral is to be interpreted in the Itô sense. Now follow the proof of Lemma 3 in [7] (in particular see equations (19)-(25)) and use (21)-(24) to get (20); (see also equations (19)-(21) in [4]).

It can be shown that [7, Eqn. (31)] (see also [4], [10])

$$\begin{bmatrix} dx(t) \\ d\hat{x}(t;\theta) \end{bmatrix} = A'(t)\begin{bmatrix} x(t)dt \\ \hat{x}(t;\theta)dt \end{bmatrix} + D'(t)u(t)dt + F'(t)\begin{bmatrix} dW_1(t) \\ dW_2(t) \end{bmatrix} \tag{A3}$$

where

$$A'(t) = \begin{bmatrix} A(t) & 0 \\ P_\theta(t)C_\theta^*(GG^*)^{-1}C(t) & A_\theta - P_\theta(t)C_\theta^*(GG^*)^{-1}C_\theta \end{bmatrix} \tag{A4}$$

$$D'(t) = \begin{bmatrix} B(t) \\ B \end{bmatrix} \tag{A5}$$

$$F'(t) = \begin{bmatrix} F(t) & 0 \\ 0 & P_\theta(t)C_\theta^*(GG^*)^{-1}G \end{bmatrix} \tag{A6}$$

and $P_\theta(t)$ satisfies the matrix Riccati equation

$$dP_\theta(t)/dt = A_\theta P_\theta(t) + P_\theta(t)\,A_\theta^* + F_\theta F_\theta^* - P_\theta(t)C_\theta^*(GG)^{-1}C_\theta P_\theta(t) \tag{A7}$$

with $\hat{x}(0;\theta) = x_\theta(0)$. Now define

$$L(t;\theta;*) = E_0\left\{\begin{bmatrix} x(t) \\ \hat{x}(t;\theta) \end{bmatrix} [x^*(t) \quad \hat{x}^*(t;\theta)]\right\} \tag{A8}$$

and use (A3) to obtain

$$\frac{dL(t;\theta;*)}{dt} = A'(t)L(t;\theta;*) + L(t;\theta;*)A'^{*}(t) + F'(t)F'^{*}(t) + D'(t)u(t)\bar{x}^{*}(t) + \bar{x}(t)u^{*}(t)D'^{*}(t) \tag{A9}$$

where

$$\frac{d\bar{x}(t)}{dt} = A'(t)\bar{x}(t) + D'(t)u(t) \tag{A10}$$

and initial conditions are

$$L(0;\theta;*) = \begin{bmatrix} P(0) & x_0 x_\theta^{*}(0) \\ x_\theta(0)x_0^{*} & x_\theta(0)x_\theta^{*}(0) \end{bmatrix} \tag{A11}$$

and $\bar{x}(0) = \begin{bmatrix} x_0 \\ x_\theta(0) \end{bmatrix}$

Finally, we have

$$P_1^0(t;\theta) = C'L(t;\theta;*)C'^{*} \tag{A12}$$

where

$$C' = [C(t) \quad -C_\theta] \tag{A13}$$

It is easy to see that

$$S(t) = [C(t) \quad 0]L(t;\theta;*) \begin{bmatrix} C^{*}(t) \\ 0 \end{bmatrix} \tag{A14}$$

Now substitute (A12) and (A14) in (20) to calculate $J(\theta)$. To calculate $J(*)$ one needs to solve a matrix Riccati equation like (A7); details are omitted since they are straightforward.

Standard numerical procedures may be employed to solve (A9). The first and second derivatives of $J(\theta)$ w.r.t. θ, needed for a gradient-type minimization procedure, can be derived as in [1]; we omit the details. Finally, sufficient conditions for the steady-state solution of (A9), hence of $J_\infty(\theta)$, may be found in [7] and [10] (see also [4]).

TABLE 1

Parameter and Approximation Coefficient Values for Example 1

Case	a	s_1, s_2	θ	K
1	0.0099	-0.01 -0.99	-0.0098	0.9996
2	0.09	-0.1 -0.9	-0.086	0.9970
3	0.16	-0.2 -0.8	-0.1489	0.9945
4	0.25	-0.5 -0.5	-0.2269	0.9909
5	0.5	-0.5 + j0.5 -0.5 - j0.5	-0.4334	0.9762
6	1.25	-0.5 + j1 -0.5 - j1	-1.0044	0.8972

s_1, s_2 - actual system poles

θ - steady-state design parameter

K - steady-state approximation coefficient

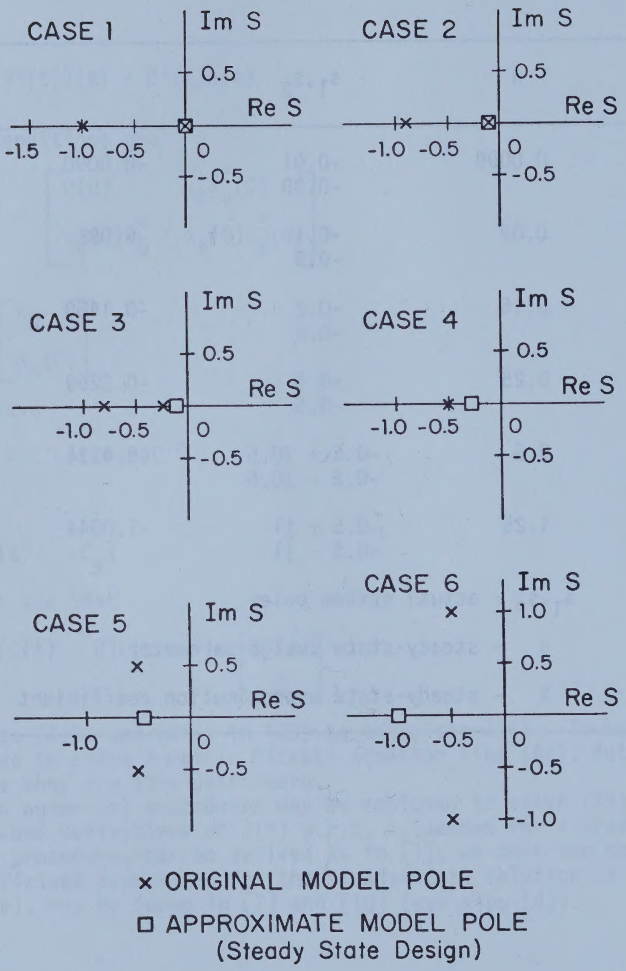

CASE 1
Im S
0.5
Re S
-1.5
-1.0
-0.5
0
-0.5
CASE 2
Im S
0.5
Re S
-1.0
-0.5
0
-0.5
CASE 3
Im S
0.5
Re S
-1.0
-0.5
0
-0.5
CASE 4
Im S
0.5
Re S
-1.0
-0.5
0
-0.5
CASE 5
Im S
0.5
Re S
-1.0
0
-0.5
CASE 6
Im S
1.0
0.5
Re S
-1.0
-0.5
0
-0.5
-1.0
× ORIGINAL MODEL POLE
□ APPROXIMATE MODEL POLE
(Steady State Design)

Fig. 1

LABELLING AND ADDRESSING IN DISTRIBUTED SYSTEMS (PRELIMINARY VERSION)

N. SANTORO, R. KHATIB
Department of Computer Science
University of Ottawa
Ottawa, Ontario K1N 9B4, Canada

ABSTRACT

We consider the problem of addressing (or routing) a message to its destination in fully decentralized networks of processors with only local topological information; each processor in the network is uniquely named, and has distinct port numbers associated to its neighbours. In these networks, no processor has explicit knowledge of the location of other processors (except the neighbours); however, if names and port numbers are assigned according to particular labelling schemes, the processors can use the information implicit in the labels to efficiently address messages. In this paper, we present labelling schemes; analyze their properties; and describe simple and highly efficient distributed addressing algorithms based on these schemes. We prove that the addressing algorithms for acyclic networks and circles of processors are optimal, and that the addressing algorithm for arbitrary networks is within a factor of two from being optimal.

I. INTRODUCTION

Consider a connected network of loosely-coupled processors where no central controller exists and only topological knowledge is available to the processors: that is, each processor is highly autonomous and is aware only of the processor to which it is directly connected (the neighbours). Every processor has a unique number (or name) and has distinct labels (or port numbers) associated to its neighbours; processors communicate by exchanging messages with their neighbours. Because of their nature, these networks are called neighbourly networks and any distributed algorithm that can be executed on a neighbourly network is called neighbourly algorithm. Examples of neighbourly algorithms can be found in [1-11, 14-16]; for a general discussion on neighbourly networks see [12].

We can represent a neighbourly network using a linear graph $G = (N,A)$ where N is the set of nodes and A is the set of arcs: each node $x \in N$ represents a processor in the network and has an associated unique name (in the following, we will denote by $n(i)$ the node having name i); and each arc $a(i,j) \in A$ represents a direct communication line between $n(i)$ and $n(j)$ (in our model, $a(i,j) = a(j,i)$). If $a(i,j) \in A$, then $n(i)$ and $n(j)$ are said to be neighbours. Each node $n(i)$ maintains a list $A(i)$ of the labels of its neighbours, and can send a message M to a subset $S \subseteq A(i)$ of its neighbours by the "send M to S" directive.

A major problem in neighbourly networks arises when processor $n(i)$ wants to send a message to processor $n(j)$ to which it is not directly connected. Since the network is connected, there is at least one path $\pi(i,j)$ between $n(i)$ and $n(j)$. However, since each processor knows only its own neighbours, processor $n(i)$ does not know a priori which neighbour is "next" in $\pi(i,j)$; the process of determining such neighbour is called addressing.

A naive solution to the addressing problem is "blind broadcasting": processor $n(i)$ sends the message to all its neighbours which, in turn, pass the message to all their neighbours, and so on; obviously, processor $n(j)$ will eventually receive the message. This scheme is very costly in terms of message exchanges: in general, O(e) messages will be exchanged to transmit a message from n(i) to n(j), where $e = |A|$.

The intrinsic difficulty of the addressing problem in neighbourly networks lies in the fact that a node has no explicit knowledge of the location of the other nodes (except its neighbours) in the network. In this paper, we show that it is possible to find a simple and efficient solution to the addressing problem while maintaining the restriction that only local topological knowledge is available at each node. This can be achieved by naming the nodes and labelling the neighbours not arbitrarily but according to some scheme. In particular, we present labelling schemes (i.e., schemes for naming the nodes and labelling the neighbours) that allow the construction of addressing mechanisms with the following properties:

1. every processor n(i), wanting (spontaneously or after receiving a request) to send a message to processor n(j), can determine a neighbour in $\pi(i,j)$ by simply checking the labels of the neighbours;

2. no processor receives the same message twice;

3. the message eventually reaches processor n(j).

In the following sections, we present labelling schemes and corresponding addressing mechanisms for acyclic networks, networks of arbitrary topology, and circles of processors. For each addressing mechanism we present a distributed (neighbourly) algorithm that implements it, and analyze its efficiency in terms of number of message exchanges and of total execution time needed to transmit a message to its destination. In particular, we prove that the addressing algorithms for acyclic networks and for circles of processors are both optimal, and that the complexity of the addressing algorithm for arbitrary networks is at most twice as the complexity of the optimal solution. The proofs of all theorems, lemmas and properties can be found in [13].

2. COMPLEXITY PARAMETERS

The complexity of a distributed algorithm is measured in terms of both of the number of message exchange and of the total execution time (i.e., the delay between the time the first processor starts the algorithm and the time the last processor is notified of its termination). Because the network may be asynchronous and time delays may be unpredictable, the total execution time is measured by assuming that the network is synchronous and that the exchange of a message requires a unit of time [12].

Let us now discuss the complexity of distributed solutions to the addressing problem. Let $M_G(i,j)$ and $T_G(i,j)$ denote the minimum number of message exchanges and of units of time needed to transmit a message in G from n(i) to n(j), respectively; then, $M(G)=\max\{M_G(i,j) \mid n(i),n(j)\in N\}$ and $T(G)=\max\{T_G(i,j) \mid n(i),n(j)\in N\}$ represent lower bounds on the number of message exchange and of units of time, respectively, needed in the worst case to transmit a message to its destination in G.

Let $d_G(i,j)$ denote the distance between n(i) and n(j) in G, and $d(G)=\max\{d_G(i,j) \mid n(i), n(j)\in N\}$ denote the diameter of G.

Then the following relations trivially hold:

Lemma 1 For all $n(i)$, $n(j) \varepsilon N$, we have

1. $M_G(i,j) \geq d_G(i,j)$
2. $T_G(i,j) \geq d_G(i,j)$

Lemma 2 For any G, we have

1. $M(G) \geq d(G)$
2. $T(G) \geq d(G)$

Given a labelling scheme L and addressing algorithm A(L) using the properties of L, let $M_G(A(L),i,j)$ and $T_G(A(L),i,j)$ denote the number of message exchanges and of time units, respectively, required by A(L) to transmit a message from $n(i)$ to $n(j)$; and let
$M_G(A(L))=\max\{M_G(A(L),i,j) \mid n(i), n(j) \varepsilon N\}$ and
$T_G(A(L))=\max\{T_G(A(L),i,j) \mid n(i), n(j) \varepsilon N\}$ be the algorithm worst case complexities.

Definition 1 An addressing algorithm A(L) is *α-efficient* for G, $\alpha \geq 1$, if

i) $M_G(A(L)) / M(G) \leq \alpha$ and

ii) $T_G(A(L)) / T(G) \leq \alpha$

Definition 2 An addressing algorithm A(L) is *optimal* for G if for all $n(i)$, $n(j) \varepsilon N$ we have

i) $M_G(A(L),i,j) = M_G(i,j)$, and

ii) $T_G(A(L),i,j) = T_G(i,j)$.

In the following sections we present optimal addressing algorithms for acyclic networks and circles of processors, and a 2-efficient addressing algorithm for arbitrary networks.

3. ACYCLIC NETWORKS

3.1 Labelling Scheme

We will first consider the case when the network is acyclic; i.e., G is a tree. Given a node $x \varepsilon N$, let $G(x) = (V(x), E(x))$ be the (rooted) directed tree constructed from G as follows:

1. set $V(x):=\{x\}; E(x):=\emptyset$;
2. for all $y \varepsilon N$ such that $(x,y) \varepsilon A$, set $V(x):= V(x) \cup \{y\}$ and $E(x) \cup \{(x,y)\}$;
3. for all $(z,w) \varepsilon A$, such that $w \notin V(x)$ and $(y,z) \varepsilon E(x)$, set $V(x):=V(x) \cup \{w\}$ and $E(x):=E(x) \cup (z,w)$.

If $(y,z) \varepsilon E(x)$, then y is called *predecessor* of z (and denoted by $p_x(z)$), and z is called *successor* of y. Let $s_x(y)$ denote the number of successors of $y \varepsilon N$ in $G(x)$; $T_x(y)$ denote the subtree of $G(x)$ rooted in y; and $t_x(y)$ denote the number of nodes in $T_x(y)$.

We assume that every node $y \varepsilon N$ can distinguish its successors in $G(x)$ and that the successors are indexed sequentially. Let $y[i]$ denote the i-th successor of y, $1 \leq i \leq s_x(y)$; "number y by a" denote the directive of assigning number a to node y; and "label y at z by a" denote the directive of assigning label a to node y in the neighbours list of node z.

The labelling scheme T for acyclic networks is described in [13]. Informally, the nodes in G are numbered according to the order they would be visited during a "post-order" traversal of $G(x)$; and the neighbours $\{y[1],\ldots,y[s_x(y)],\ p_x(y)\}$ of a node $y \varepsilon N$ are labelled according to the following criteria:

(i) the label of $p_x(y)$ is ε;

(ii) the label of $y[i]$, $1 \leq i \leq s_x(y)$, is the smallest number given to the nodes in the subtree $T_x(y[i])$ of $G(x)$ rooted in $y[i]$.

Let $f(v)$ denote **the number** given to node $v \varepsilon N$ by labelling scheme T ; and $g(v,z)$ denote the label given to neighbour z of v by labelling scheme T. For completeness sake, let us assume that $f(x,p_x(x))$ is defined and $f(x,p_x(x))=1$.

Lemma 3
1. $f(x) = |N|$;
2. $g(x, x[1]) = 1$;
3. $\forall v \varepsilon N$, if $v \neq x$ then $g(v,p_x(v)) = \varepsilon$.

Property 1 Let $v[i]$ be a successor of $v \varepsilon N$, $1 \leq i \leq S_x(v)$ - Then the following relations hold.

1. $g(v,v[i]) = \begin{cases} g(v,v[i-1]) + t_x(v[i-1]) & \text{if } i>1 \\ g(p_x(v),v) & \text{otherwise} \end{cases}$
2. $f(v[i]) = g(v,v[i]) + t_x(v[i]) -1$

Property 2 Let $v \varepsilon N$. Then the following relations hold.

1. If $S_x(v) \geq 1$, then, for all i, $1 \leq i \leq S_x(v)$,

   $f(v) > f(v[i]) \geq g(v,v[i]) \geq g(p_x(v),v)$.
2. If $S_x(v) = 0$, then $f(v) = g(p_x(v),v)$

Property 3 Let $v \varepsilon N$ such that $s_x(v) \geq 1$, and let z be a node in $T_x(v[i])$, $1 \leq i \leq S_x(v)$.
Then the following relations hold

1. $g(v,v[i]) \leq f(z) < f(v)$
2. if$<i$ $S_x(v)$ then $f(z) < g(v,v[i+1])$

3.2 Addressing Mechanism

Let us now discuss an addressing mechanism which employs the proposed labelling scheme.

Consider the situation when node $n(i)$ want to send a message to node $n(j)$; the only information available to $n(i)$ are the names i and j, and the labels associated to its neighbours. Let $A(i) = <i_1, i_2,\ldots, i_k>$ be the list of the label of the neighbours of $n(i)$; without loss of generality, assume $i_1<i_2<\ldots<i_{k-1}$ and $i_k=\varepsilon$. Let $\pi(i,j)$ denote the shortest path in G between $n(i)$ and $n(j)$; obviously, $|\pi(i,j)| = d(i,j)+1$, where $|\pi|$ denotes the lenght of path π.

Property 4 Let $b(i,j) \varepsilon A(i)$ be defined by

$$b(i,j) = \begin{cases} i_k & \text{if } j>i \text{ or } j<i_1 \\ i_s & \text{otherwise} \end{cases}$$

where s is the index such that $i_{s-1}<j\le i_s$. Then, $b(i,j) \varepsilon \pi(i,j)$

In other words, by just checking the labels associated to its neighbours, a node can directly and uniquely determine to which of its neighbours a message should be sent to reach its destination.

Let us now present the distributed addressing algorithm for acyclic networks.

```
Algorithm A(T)
1.  Node n(i) want to send a message M to node n(j)
    begin
       if (j>i or j<$i_1$) then send (i,j,M) to $i_k$
       else
       begin
           find s such that $i_{s-1}<j\le i_s$;
           send (i,j,M) to $i_s$;
       end;
    end;

2.  Node n(p) receives (source, sink, message) from a neighbour.
    begin
       if p = sink then process message
       else
           if (sink >$p_{s_x(p)}$ or sink < $p_1$) then
               send (source, sink, message) to $p_{dg(p)}$
           else
           begin
               find s such that $p_{s-1}<j\le p_s$;
               send (source, sink, M) to $p_s$;
           end;
    end;
```

where $dg(p) = S_x(p)+1$ is the degree of node $n(p)$.

Theorem 1 Algorithm A(T) is optimal for acyclic networks

4. NETWORKS OF ARBITRARY TOPOLOGY

If G is not a tree, we can still use the results of the previous section by constructing a spanning-tree of G and then applying to it the algorithm for acyclic networks. Unfortunately, the resulting addressing is no longer optimal. However, by carefully choosing the spanning-tree, the resulting addressing algorithm can still be quite efficient.

Given $x \varepsilon N$, let $r(x) = \max \{d_G(x,y) \mid y \varepsilon N\}$; then, a node $c \varepsilon N$ is said to be a center of G if for all $x \varepsilon N$ $r(c) \le r(x)$. If c is a center of G, then $r(c)$ is called radius of G. For the centers of G, the following property holds:

Lemma 4 For all $n(i)$, $n(j) \varepsilon N$,
$r(c) \le d_G(i,j) \le 2 \cdot r(c)$

Therefore, we can design a labelling scheme for arbitrary networks as follows:

```
Labelling Scheme AG
begin
    construct a minimum-distance spanning-tree
    S(G) rooted in a center of G;
    apply to S(G) the labelling scheme for acyclic networks;
    label the neighbours in G but not in S(G) with consecutive integers
    from n+1 to e.
end;
```

Then, the addressing algorithm A(AG) for an arbitraty network G will be the addressing algorithm A(T) applied to S(G), and will have the following properties:

Property 5 For all n(i), n(j) εN we have

$$M_G\ (A(AG)) \le 2 \cdot r(c).$$

Theorem 2 Algorithm A(AG) is 2-efficent for arbitrary graphs

5. CIRCLE OF PROCESSORS

We will now consider the special case when the processors are loosely-coupled in a circular arrangement. In this case, every node x has two neighbours, usually labelled "left" and "right". Choosen an arbitrary global direction in the circle, let R(x) and L(x) denote the neighbour to the right and to the left of node $x \varepsilon N$, respectively. For any integer S, let $R^S(x) = R(R^{S-1}(x))$ and $L^S(x) = L(L^{S-1}(x))$, where $R^0(x) = L^0(x) = x$; obviously, $R^S(x) = L^{n-S}(x), 0 \le S \le n$.

Consider the following labelling scheme for circles.

```
Labelling Scheme C
begin
    choose an arbitray $x \varepsilon N$;
    for S=0 until n-1 do
    begin
        name $R^S(x)$ by s+1;
        label $R^{S+1}(x)$ at $R^S(x)$ by "right";
        label $L(R^S(x))$ at $R^S(x)$ by "left";
    end;
end;
```

Let us now discuss the addressing mechanism. Consider the situation when n(i) wants to send a message to n(j); the only information available at n(i) are the names i and j and the labels "right" and "left". Let $\bar{\pi}(i,j)$ denote the shortest path between n(i) and n(j); obviously $|\bar{\pi}(i,j)| = d(i,j)+1$.

Property 5 Let b(i,j) be defined by

$$b(i,j) = \begin{cases} \text{"right"} & \text{if } [i-j]_{\text{mod } n} > [j-i]_{\text{mod } n} \\ \text{"left"} & \text{otherwise} \end{cases}$$

Then $b(i,j) \varepsilon \bar{\pi}(i,j)$.

The corresponding addressing algorithm is described below.

Algorithm A(C)

```
1. Node n(i) want to send a message M to node n(j)
   begin
       if $[i-j]_{\mathrm{mod}\ n} > [j-i]_{\mathrm{mod}\ n}$ then
           send (i,j,M) to "right"
       else
           send (i,j,M) to "left"
   end;

2. Node n(j) receives (source, sink, message) form "left" ["right"]
   begin
       if p = sink then process message
       else send (source, sink, message) to "right" ["left"]
   end ;
```

Theorem 3 Algorithm A(C) is optimal for circles of processors.

6. CONCLUDING REMARKS

In this paper, we have presented labelling schemes and efficient addressing algorithms for networks of processors where only local topological knowledge is available at each processor. Since the labelling process is typically done at design time, we have given only a "centralized" description of the labelling schemes. However, decentralized labelling algorithms implementing the schemes can be easily devised using a combination of the existing algorithms for numbering undistinguishable nodes [1,15], and for electing a leader [2-5,7,8, 10,11,14], finding the center [9], or constructing a spanning-tree [5,15] in a network of processors with distinct names. In particular, the labelling can be done in a distributed way exchanging $O(n)$ messages in acyclic networks, $O(n\ e)$ messages in arbitrary networks, and $O(n.\log_2 n)$ in circles of processors.

REFERENCES

[1] Aburdene, M.F. Numbering the nodes of a graph by distributed algorithms, Proc. 17th Allerton Conf. on Communication, Control, and Computing, 1979, 634-639.

[2] Angluin, D. Local and global properties in networks of processors. Proc. 12th ACM Symp. on Theory of Computing, 1980, 82-93.

[3] Burns, J.E. A formal model for message-passing systems. Tech. Rep.91, Computer Science Department, Indiana University, 1980.

[4] Chang, E.J., and Roberts, R. An improved algorithm for decentralized estrema-finding in circular configurations of processes. Comm. ACM 22, 5 (May 1979), 281-283.

[5] Gallager, R.G. Finding a leader in a network with 0(E) + 0(N,log N) messages. MIT Internal Memorandum, 1979.

[6] Gallager, R.G., Humblet, P., and Spira, P. A distributed algorithm for minimum weight spanning trees. Tech. Rep. LIOS-P-o06A, MIT, Oct. 1979.

[7] Hirschberg, D.S. Election processes in distributed systems. (Summary) Proc. 18th Allerton Conf. on Communication Control and Computing, 1980, 823.

[8] Hirschberg, D.S., and Sinclair, J.B. Decentralized extrema-finding in circular configurations of processes. Comm. ACM 23, 11 (Nov. 1980), 627-628

[9] Korach, E., Rotem, D., and Santoro, N. Efficient distributed algorithms for finding centers and medians in a network. Proc. 18th Allerton Conf. on Communication, Control and Computing, 1980, 687-696.

[10] Korach, E., Rotem, D., and Santoro, N. A probabilistic algorithm for extrema-finding in a circle of processors. Tech. Rep. CS-81-19, Dept. Computer Science, University of Waterloo, 1981.

[11] Lelann, G. Distributed systems-toward a formal approach. Inform. Proc. 77, 155-160.

[12] Santoro, N. Distributed algorithms for very large distributed environments: new results and research directions. Proc. CIPS 81, 141-145.

[13] Santoro, N., and Khatib, R. Labelling schemes and addressing mechanisms in distributed systems. Tech. Rep. TR-91-10, Computer Science Department, University of Ottawa, 1981.

[14] Santoro, N., Korach, E., and Rotem, D. Decentralized extrema-finding in circular configurations of processors: an improved algorithm. Proc. 11th Conf. on Numerical Mathematics and Computing, 1981, (to appear).

[15] Soloway, S.R. On the node-numbering problem. Proc. 18th Allerton Conf. on Communication, Control and Computing, 1980, 514-523.

[16] Spira, P.M. Communication complexity of distributed minimum spanning tree algorithms. Proc. 2nd Berkeley Conf. on Distributed Data Management and Computer Networks, 1977.

DESIGNING EFFICIENT PARALLEL ALGORITHMS FOR MULTIPLE-OUTPUT COMPUTATIONS WITH FAN-IN CONSTRAINTS

(Preliminary Version)

NICOLA SANTORO, ROBERT L. PROBERT
Department of Computer Science
University of Ottawa
Ottawa, Ontario
K1N 9B4 Canada

ABSTRACT

In this paper we investigate natural "fan-in" type limitations to processor utilization in multiple output computations. Using thie limitation, we establish lower-bounds for simple-input multiple-output problems and show the existence of a computational duality between these and multiple-input single-output problems. A class of standard composite cost measures are studied, and an argument is presented that these measures are biased or inappropriate for this class of parallel computations. To correct this bias, a new class of cost measures is proposed.

INTRODUCTION

The design and analysis of algorithms to be executed by a serial processor has been a fertile area of research over the past decade. Comparatively, the state of knowledge (in particular, empirical results based on practical experience) in parallel computation is still primitive. Although a considerable body of theoretical work has been developed, there is a definite need for new techniques for the design and analysis of "efficient" parallel algorithms. In particular, there is an urgent need to reconsider the cost measures traditionally used to determine the efficiency of parallel algorithms.

In this paper we show that there is a large class of computations, which we refer to as constrained computations, for which traditional cost measures are inappropriate. These measures are unweighted functions of (one or more of) parallel time, number of processors, and serial time; and will be shown to be inherently biased toward sequenciality for this class of computations. Therefore, the evaluated efficiency of a parallel algorithm, with respect to these measures, will reflect more this bias than the complexity of the problem or function being computed.

In the following, we first investigate natural "fan-in" type limitations to processor utilizations in simple-input multiple-output problems, and show the existence of a computational duality between these and multiple-input single-output problems. Evidence that traditional cost measures are biased toward sequenciality is presented in the second part of this paper.

For expository purposes, details of proofs are omitted; the interested reader will find these in [8].

BASIC DEFINITIONS AND RESULTS

Our model of computation is relatively standard [1,2,9]. In particular, in our use of this shared-memory model, we do not concern ourselves with details of input, output, storage, and retrieval operations. We can characterize the class of problems that we are interested in solving, as triples (I,O,B) where I is a non-empty set of inputs, O is a non-empty set of outputs, and B is a non-empty set of binary operations of similar scope such as $\{*,/\}$, which can be performed by all the processors. A problem (I,O,P) is k-computable in time t if k processors working together can compute all the elements in O in t parallel steps. Unbounded parallelism refers to the case in which there is not a-priori bound on k; i.e., $k = \infty$. Many parallel algorithms have been developed and studied for the computation of multiple-input simple-output (MISO) problems; that is, problems where $I \geq 1$ and $|O| = 1$. Polynomial evaluation, inner product and evaluation of Boolean expressions are a few examples of this class of problems. Lower bounds on the number of parallel steps required to solve MISO problems have been established by Kung [6] and by Munro and Paterson [7]. The latter is based on serial processing time as well as a "fan-in argument", and is reported here in a slightly modified version.

Let C(Q,k) denote the number of active operations required by the shortest k-computation of Q = (I,O,B), where active operations are those operations which do not involve scalar operands [1].

Lemma 1: Given an MISO problem Q, let $C(Q,1) = q$, $q \geq 1$.

Then

$$C(Q,k) \geq \begin{cases} \lceil (q+1-2^{\lceil \log k \rceil})/k \rceil - \lceil \log k \rceil & \text{if } q \geq 2^{\lceil \log k \rceil} \\ \lceil \log\ (q+1) \rceil & \text{otherwise} \end{cases} .$$

The proof of this lemma was based on the observation that there exists a "fan-in constraint" for the evaluation of MISO problems; namely, every algorithm computing a MISO problem Q will be usefully employed at most one processor in the last computation step; at most two processors in the previous step, and so forth. In the rest of this section, we will show that a similar kind of constraints exists for a different class of problems.

A problem Q = (I,O,B) is simple-input multiple-output (SIMO) if $|I| = 1$ and $|O| \geq 1$. A typical SIMO problem is the computation of the first n powers of x using only multiplications. Munro and Paterson [7] gave an algorithm for n processors which was optimal in time and number of operations performed. Borodin and Munro [1] observed that n/2 processors sufficed, yielding

Lemma 2: Let $Q = (\{x\},\{x,x^2,\ldots,x^n\},\{*\})$. Then, Q is (n/2)-computable in time $\lceil \log n \rceil$ with only n-1 operations being performed. The time and number of operations are optimal.

The proof of the optimality of the algorithm follows from a more general lower-bound argument given by Kung and reported in [1]. This argument is a straightforward induction on limits to the rate of growth of intermediate results in the computation in F[x], given $F \cup \{x\}$, of a rational function of given degree.

As a preliminary step in establishing a lower-bound on SIMO problems, we will prove a result for the above problem that is more general than the above lemma.

Theorem 1: Let $Q = (\{x\},\{x,x^2,\ldots,x^n\},\{*\})$. Then Q is k-computable in time

$$\lceil (n-2^{\lceil \log k \rceil})/k \rceil + \lceil \log k \rceil$$

with only n-1 operations being performed. Furthermore, this is optimal for $k < n$.

The proof of this theorem involves, first of all, designing an algorithm which maximizes processor utilization to obtain the upper-bound, and then employing essentially a rate of growth argument similar to the above to demonstrate the lower-bound. Thus, there exists for this problem a constraint similar to the "fan-in constraint" found in MISO problems. This "fan-out constraint" is actually characteristic of all SIMO problems as proved by the following result.

Theorem 2: Given an SIMO problem Q, let $C(Q,1) = q$, $q \geq 1$. Then

$$C(Q,x) \geq \begin{cases} \lceil (q + 1 - 2^{\lceil \log k \rceil})/k \rceil + \lceil \log k \rceil & \text{if } q \geq 2^{\lceil \log k \rceil} \\ \lceil \log(q+1) \rceil & \text{otherwise.} \end{cases}$$

Theorem 2 establishes a (perhaps not too surprising) computational duality between MISO and SIMO problems. As an aside, Theorem 2 yields a lower-bound for the parallel computation of addition-sequence problems; i.e., problems of the form $(\{1\},\{x_1,x_2,\ldots,x_n\},\{+\})$ [3,4].

CONSTRAINED PROBLEMS AND BIASED COST MEASURES

As a consequence of the results presented in the previous section, we can treat MISO and SIMO problems as belonging to the same class of "constrained problems". We will now analyze the implications of the previous results on the design of parallel algorithms.

The first consideration in designing an efficient parallel algorithm must be a precise definition of the cost measures. In the literature, several cost measures have been presented; the most important ones are listed below, where P denotes the resources (i.e., the number of processors), T denotes the time (i.e., the number of parallel steps), and T_1 denotes the best serial time.

i) Computing Capacity (i.e., time/resource product):
$C = T \times P$;

ii) Efficiency (i.e., best serial time/computing capacity ratio):
$E = T_1/(T \times P)$

iii) Speed-up (i.e., best serial time/time ratio):
$S = T_1/T$.

To simplify the following discussion, we will restrict ourselves only to constrained problems where $n = \max\{|I|,|O|\}$ is a power of two and $T_1 = n-1$. Incidentally, a large number of constrained problems have these characteristics: determining the sum (or product) of n numbers, finding the max (or min) of n elements, multiplying two (n/2)-vectors, computing the first n powers of x, etc. This restriction can be easily lifted to allow a generalized treatment of all constrained problems. Let Q be a constrained problem satisfying the above assumption.

Lemma 3: Let $P = \alpha(n/\log n)$, $\log n \geq \alpha \geq 0$. Then $T \geq (1 + (1/\alpha)) \log n + \log \alpha - \log \log n - 1$.

An immediate consequence of this lemma is the following.

Lemma 4: Let $Q = (\{x\},\{x,x^2,\ldots,x^n\},\{*\})$, and $n = 2^p$, $p > 1$. Then Q is (n/log n)-computable in time $2 \log n - \log \log n$ with only n-1 operations being performed. Furthermore, this is optimal.

Let us now compare the results of Lemmas 2 and 4. Using Lemma 2 we have $P = n/2$ and $T = \log n$. Therefore, the computing capacity C', the efficiency E' and the speed-up S' are

$$C' = \frac{1}{2} n \log n$$

$$E' = \frac{2(n-1)}{n \log n}$$

$$S' = \frac{n-1}{\log n} \quad .$$

Using Lemma 4, we have $P = n/\log n$ and $T = 2 \log n - \log \log n$. Therefore, the computing capacity C", the efficiency E", and the speed-up S" are

$$C'' = 2n - \frac{n \log \log n}{\log n}$$

$$E'' = \frac{(n-1)\log n}{n(2 \log n - \log \log n)}$$

$$S'' = \frac{n-1}{2 \log n - \log \log n}$$

It is easy to verify that, for $n \geq 4$, $C'' < C'$, $E'' > E'$, and $S'' > S'$. That is, if we use more than n/log n processors, we will actually decrease the efficiency of the algorithm.

The quantity n/log n has been observed by several individuals, including one of the authors, to be an optimal allocation unit for processors for a wide variety of standard problems [5]. What has not been observed is the fact that if we use less than n/log n processors, then our algorithms will become more efficient. This can be easily seen in Table I for values of P of the form $\alpha(n/\log n)$. Note the apparent existence of a "stepwise cost coefficient" which characterizes tradeoff steps for computation capacity: if α is an integer, it gives integer coefficients in the leading term of C; if α is the inverse of an integer, it yields an integer coefficient in the leading term of T.

To return to our observation, C is a decreasing function in P, while E and S are increasing functions in P. Moreover, because of the equivalence between MISO and SIMO problems, this result will hold also for the problems of summing n numbers, finding the min of n elements, multiplying two (n/2)-vectors, etc.

We interpret this result to mean that the traditional cost measures (such as computation capacity, efficiency, and speed-up) are often inappropriate measures for parallel computations, since they exhibit a bias toward serial computation. This is mostly due to the fact that both C and E give the same "importance" to P and T, and that S gives the same "importance" to the serial time and to T. Therefore, to correct this bias, we propose the use of cost measures in which time is considered a more important factor than the serial time or the amount of used resources. For example, an alternative to computation capacity could be the measure $T^2 \times P$. It is interesting to note that this measure has a minimum when $P = n/\log n$, as shown in Table II. This proposal has the advantage not only of correcting the bias, but also of reflecting the implications that the recent advances in "chip" technology have on "efficiency trade-offs". Thus, for example, a chip could be constructed from n/log n processors which would never perform (constrained) computations more slowly than half the optimal speed.

P	T*	C*	E*	S*
n	log n	n log n	$\frac{(n-1)}{n \log n}$	$\frac{n-1}{\log n}$
⋮	⋮	⋮	⋮	⋮
$\frac{2n}{\log n}$	1.5 log n	3n	$\frac{n-1}{3n}$	$\frac{n-1}{1.5 \log n}$
⋮	⋮	⋮	⋮	⋮
$\frac{n}{\log n}$	2 log n	2n	$\frac{n-1}{2n}$	$\frac{n-1}{2 \log n}$
⋮	⋮	⋮	⋮	⋮
$\frac{n}{2 \log n}$	3 log n	1.5 n	$\frac{n-1}{1.5\ n}$	$\frac{n-1}{3 \log n}$
⋮	⋮	⋮	⋮	⋮
1	n-1	n-1	1	1

Table I. Cost measures for $(\{x\},\{x,x^2,\ldots,x^n\},\{*\})$.

*Only the leading term is shown.

P	T*	$T^2 \times P^*$
n	$\log n$	$n \log^2 n$
⋮	⋮	⋮
$\frac{2n}{\log n}$	$1.5 \log n$	$4.5\, n \log n$
⋮	⋮	⋮
$\frac{n}{\log n}$	$2 \log n$	$4\, n \log n$
⋮	⋮	⋮
$\frac{n}{2 \log n}$	$3 \log n$	$4.5\, n \log n$
⋮	⋮	⋮
1	$n-1$	n^2

Table II. New cost measure for $(\{x\},\{x,x^2,\ldots,x^n\},\{*\})$.

*Only the leading term is shown.

REFERENCES

[1] A. Borodin, J. I. Munro, The Computational Complexity of Algebraic and Numeric Problems. Elsevier Pub. Co., New York, 1975.

[2] R. P. Brent, "The Parallel Evaluation of Arithmetic Expressions in Logarithmic Time," Symp. on Complexity of Sequential and Parallel Numerical Algorithms.

[3] D. Dobkin, R. J. Lipton, "Addition Chain Methods for the Evaluation of Specific Polynomials," SIAM J. Comp. 9 (1980), 121-125.

[4] P. Downey, B. Leong, R. Sethi, "Computing Sequences with Addition Chains," SIAM J. Comp. 10 (1981), 638-646.

[5] D. S. Hirschberg, A. K. Chandra, D. V. Sarwate, "Computing Connected Components on Parallel Computers," CACM 22 (1979), 461-464.

[6] H. T. Kung, "Some Complexity Bounds for Parallel Computation," 6th ACM Symp. on Theory of Computing, 1974, 323-333.

[7] I. Munro, M. Paterson, "Optimal Algorithms for Parallel Polynomial Evaluation," JCSS 7 (1973), 189-198.

[8] R. L. Probert, N. Santoro, "Designing Efficient Parallel Algorithms for Constrained Computations," Comp. Sci. Dept. Tech. Rep. TR-81-14, University of Ottawa, 1981.

[9] H. S. Stone, "Problems of Parallel Computation," Symp. on Complexity of Sequential and Parallel Numerical Algorithms, 1973, 1-16.

A PARALLEL COMPUTATION OF CONNECTED COMPONENTS

G. D. LAKHANI
Computer Science Section/Department of Electrical Engineering
Texas Tech University
Lubbock, Texas

ABSTRACT

An algorithm is presented that computes the connected components of n-node, undirected graphs on n parallel processors. It runs in linear time and improves the time-processor product bound of the previously best known algorithm by a factor of O(lg n).

INTRODUCTION

In this note, we consider the problem of parallel computation of connected components of undirected graphs. Much work on this problem has been done recently [2,3,4]. The algorithm of Hirschberg, Chandra and Sarwate [2] runs in time $O(\lg^2 n)$ for graphs with n vertices and is designed for the shared memory model of SIMD machines with n^2 processors. (Actual time-processor product bound is $O(n^2 \lg n)$). An implementation of this algorithm for graphs with n^q vertices (each of degree $\leq d$) on mesh-connected network of n^q processors is shown in [3] by Nassimi and Sahni. The implementation runs in time $O(q^3(q+d)n \lg n)$. Reghbati and Corneil have also obtained an algorithm which has much higher bound [4]. Here, we shall present an algorithm that runs in O(n) time and uses n processors, thus has $O(n^2)$ bound.

Very little is said in [2] about the model of computation. We use the same model now commonly known as the shared memory model of SIMD machines. We assume that the n processors of the machine share one common memory, that all n processors can fetch n distinct memory words in the same time slot but concurrent access of any word by two or more processors for either reading or writing is not permitted, that each processor may have some (fixed) storage. This model has been used in variety of parallel computation problems. For a big list, see [1] and references [4,5,8, 9,13,14] in [1] .

Let V = {1,2,...,n}, and let G = (V,E) be an undirected graph with vertices in V and edges in E. Furthermore, let D be the adjacency matrix of G so that D(i,j) = 1 if an edge exists between i and j, or else D(i,j) = 0. A connected component is a maximal subgraph such that every two vertices in the subgraph are joined by a path. A fragment is a connected subgraph. Two fragments are said to be adjacent to each other if there exist two vertices in the two fragments which are joined by an edge in G.

We present the algorithm and its correctness in the next section and its analysis in the following section.

THE ALGORITHM

Our strategy is close to the one used by Hirschberg et al. The essential difference is in the order and the method that we use for combining fragments. In our approach, we consider each vertex to be in some fragment (initially, each vertex is alone), compute a 'closest' fragment for every fragment in parallel, connect fragments to closest fragments serially and update the matrix D in parallel. We use certain notations that we explain here. CONNECT(i) is a list containing the vertices that are known to be connected with i. S is the set of fragments for the current iteration of the main loop. COMP collects the connected components. We use D(i,*) and D(*,i) to denote the i^{th} row and i^{th} column of D, respectively. lg n denotes $\log_2 n$. or is a parallel operation that computes component-wise logical 'or' of two vectors of equal size. Algorithm is given below.

```
0. Let S = {1,2,...,n}, COMP=∅.
1. For each i in S do CONNEC(i) = {i}.
2. Repeat steps 3-7 ⌈lg n⌉ times while (S=∅).
3. For each i in S do
     select the smallest j in S such that D(i,j) = 1.  In
     case no such j exists, delete i from S and add it to
     COMP.
4. Sort the set {(i,j) : i in S and j selected for i in
     step 2} so that (i,j) ≥ (i',j') if either ij>i'j', or
     ij = i'j' and j ≥ j'.
5. For each pair (i,j) in the set, varying it from the
     highest to the lowest, compute
     D(j,*) = or {D(i,*), D(j,*)}.
6. For the same sequence of pairs (i,j) as in step 4, compute
     D(*,j) = or {D(*,i), D(*,j)},
   delete i from S and append CONNECT(i) to CONNECT(j).
7. For each i in S do D(i,i) = 0.
```

We shall describe the actions of various steps of the algorithm. We shall verify that every element of S is a fragment of graph G, and that D maintains adjacency between fragments in S. We identify a fragment by the smallest vertex in it which we call the root of the fragment. Initially, every vertex is a fragment, and therefore, the two statements about S and D are true for the initial iteration.

Assume that the statements about S and D are true for some iteration. Step 3 computes an adjacent fragment for every fragment with which the fragment will combine. Step 4 is designed to determine an order for connecting fragments. The order is such that pair (i,j) precedes (j,k), i.e. all fragments i that need be connected to the fragment j (selected for i) are connected before j is connected to the fragment k chosen for j in step 2. The connections are made in step 5. As we process a pair (i,j), we connect the fragment i to j by selecting any edge between some two vertices in the two fragments, (the combined fragment is now denoted by j); and we compute, by taking component-wise 'or' of the i^{th} and j^{th} rows of D, the adjacency of every fragment in S with respect to (new) j. The i^{th} row can now be ignored for the rest. Thus, by the end of step 5, each unignored row of D stores adjacency between a fragment that will belong to S in the next iteration of the main loop and the fragments that were present in S at the beginning of the current iteration. Step 6 updates D by taking component-wise 'or' of the columns which store adjacency of the fragments that have been combined into one fragment in the previous step. These columns can also be ignored. The actions of step 5 and 6 verify the correctness of the statements about S and D. Step 7 is designed to prevent a fragment i from being connected to itself in remaining iterations.

The correctness of the algorithm is now obvious. COMP clearly picks up only those fragments which have no more adjacency, and therefore, are the connected components of G. For every i in COMP, CONNECT(i) denotes the vertices that are connected with i. The condition S = ∅ is bound to attain in ⌈lg n⌉ iterations as the number of elements in S reduce by at least a factor of 2 in every iteration.

THE ALGORITHM IS OPTIMAL

The analysis of the algorithm is quite simple. Each of its steps can be processed in parallel. Step 3 is computed, by finding the component-wise 'or' of the unignored columns of D, in time $O(|S|)$. A number of sorting routines are available for step 4 which run in time $O(\lg^2 n)$. See [1]. The time for this step is, therefore, $O(\lg^2 |S|)$. Step 5 performs $|S|$ or operations on the unignored rows of D, and step 6 performs on the unignored columns of D. Both take $O(|S|)$. Step 7 can be done in $O(1)$ time. Thus, the total time required is

$$\sum_{k=0}^{\lceil \lg n\rceil -1} C\,|S| \le C \sum_{k=0}^{\lceil \lg n\rceil -1} n/2^k \le 2\,Cn = O(n),$$

where C is a constant. Thus, we have a linear time algorithm for which the time-processor product bound is of order n^2. We shall call the algorithm to be optimal because any optimal, uniprocessor, algorithm will take $O(n^2)$ time in the worst-case.

We shall now describe the essential differences between our algorithm and Hirschberg's algorithm [2] that cause improvement in the time-processor product bound. An obvious difference is that we delete a vertex (or fragment) once it has been connected to some fragment and its adjacencies have been absorbed in the root vertex whereas all the adjacencies of every vertex is checked in every iteration in their algorithm. Another difference is in the order that we use for combining fragments. We combine two fragments at a time whereas all the connections of fragments (they are called supervertex) are computed in parallel in their algorithm and, therefore, they have some redundant computations. We are able to combine serially and still keep the time low because the number of fragments reduce by at least a factor of 2.

NOTE: We have applied this algorithm for a number of related problems. One problem that of computing transitive closure [2] can be solved concurrently by our algorithm.

REFERENCES

1. D. S. Hirschberg, "Fast Sorting Algorithms," Comm. A.C.M. Vol. 21, pp 657-661, 1978.
2. D. S. Hirschberg, A. K. Chandra and D. V. Sarwate, "Computing Connected Components on Parallel Computers," Comm. A.C.M., Vol. 22, pp 461-464, 1979.
3. D. Nassimi and S. Sahni, "Finding connected components and connected ones on a Mesh-Connected Parallel Computers," SIAM J. Comput., Vol. 9, pp 744-757, 1980.
4. E. Reghbati and D. G. Corneil, "Parallel Computation in Graph Theory," SIAM J. Comput., Vol. 2, pp 230-237, 1978.

A PARALLEL ALGORITHM FOR DETERMING CONVEX HULLS OF SETS OF POINTS IN TWO DIMENSIONS*

ANITA L. CHOW
GTE Laboratories, Incorporated
Waltham, Massachusetts

ABSTRACT

The problem of convex hulls arises in many applications: cluster anlysis, numerical analysis, and image processing. In this paper, a parallel algorithm for finding the convex hull of a set of N points in the Euclidean plane is presented. The running time required is $O((\log N)^2)$ with N processors and is $O(1/\alpha \log N)$ with $N^{1+\alpha}$ processors ($0 < \alpha \leq 1$). The models of computation used in this paper are the shared memory model and the cube-connected-cycles model. The validity of the first machine resides in uncovering the inherent data-dependence of the problem, while the second, which complies with the VLSI technological constraints, is appropriate for the development of practical parallel algorithms.

INTRODUCTION

The existence of parallel computers has motivated the development of parallel algorithms for solving many problems, both numerical and non-numerical. One area that has not been explored, however, is the use of parallel computation to solve geometric problems. Traditionally, these problems have been solved using a uniprocessor system [1, 4, 6]. The purpose of this paper is to explore the use of parallel computation in solving a particular type of geometric problem, the determination of convex hulls in two dimensions.

The problem of convex hulls arises in many applications, such as finding the diameter of a set. Typically, these applicaitons will be found in engineering disciplines such as image processing. Fast and efficient algorithms for determining convex hulls would be of great value to these disciplines.

In formal terms, the convex hull (CH) of a finite set S of points is the intersection of all convex sets containing S. In the plane, the convex hull of S, CH(S), is a convex polygon. Several optimal algorithms for determining sequentially the convex hull of a set of N points in two dimensions have been developed [1,4]. However, these algorithms achieve the running time of O(NlogN), which is not sufficiently fast for solving

---

* This work was supported in part by the National Science Foundation under Grant MCS 78-13642 and in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U.S. Air Force) under contract N00014-79-C-0424.

large problems.

The sequential algorithms use the well-known technique of "divide and conquer." This technique can also be used in parallel computation, and the subproblems generated can be solved simultaneously. However, an efficient algorithm for combining the results of these subproblems is essential for an overall fast parallel algorithm.

This paper describes a method for determining, in parallel, a convex hull of N points. It will be shown that this method can be used to develop algorithms with reference to two models of parallel computation: the Shared Memory Model (SMM) [3], and the Cube-Connected-Cycles (CCC) Model [5].

PRELIMINARIES

In this section, terms and concepts crucial to our discussion will be defined.

To begin with, a simple polygon is in standard form if its vertices occur in clockwise order with all verices distinct and no three consecutive vertices collinear. The ordering of vertices begins with the vertex that has the largest y-coordinate. This vertex will be indexed as '0'. Polygons will be expressed as sequences of vertices beginning with 0 and ending with the vertex n-1, where n is the number of points comprising the polygon. The notation for such a ploygon will be A(0:n-1).

Given a convex polygon A in standard form,

ℓ_A will be the index of the vertex with the least x coordinate

r_A will be the index of the vertex with the largest x coordinate

s_A will be the index of the vertex with the least y coordinate

The polar angle of a vertex i in the polygon, with vertex i-1 as the origin, will be defined as $\alpha_{i-1,i}$(1). Due to convexity, in the range $0 < i \leq n-1$, the sequence $(\alpha_{01}, \ldots, \alpha_{i-1,i}, \ldots)$ will be decreasing.

Next, we shall consider the properties of two non-intersecting convex polygons, A(:n-1) and B(0:m-1), where the y-coordinate of A(i) is less than that of B(j). Every point A(i) on A will generate a sequence of angles when a line is drawn from that point to some point B(j) on B. The convex angle formed by this line and the x-axis will be defined as $\gamma_{i,j}$(2).

---

(1) $\alpha_{i-1,i}$ is defined as an angle for explanatory purposes only; in the implementation of the operation of computing two angles, we shall avoid the computation of angles by replacing it with the operation of compaing the negative values of their contagents, where the function cotangent: $[0, \pi] \dashrightarrow [-\infty,\infty]$ is an order-reversing mapping.

(2) Same as (1)

A sequence is V-bitonic if it consists of a decreasing sequence, which may be empty, followed by an increasing sequence. Due to convexity, in the range $0 \le i \le s_A$, the sequence $(\alpha_{i,0},\alpha_{i,1},\ldots,\alpha_{i,s_B})$ is V-bitonic.

A sequence is Λ-bitonic if it consists of an increasing sequence, which may be empty, followed by a decreasing sequence. Due to convexity, in range $s_A \le i < n$, the sequence $(\gamma_{i,s_B},\gamma_{i,s_B + 1},\ldots,\gamma_{i,m})$ is Λ-bitonic.

Figure 1 illustrates an example of a V-bitonic sequence $(\gamma_{i,0},\ldots\gamma_{i,s_B})$.

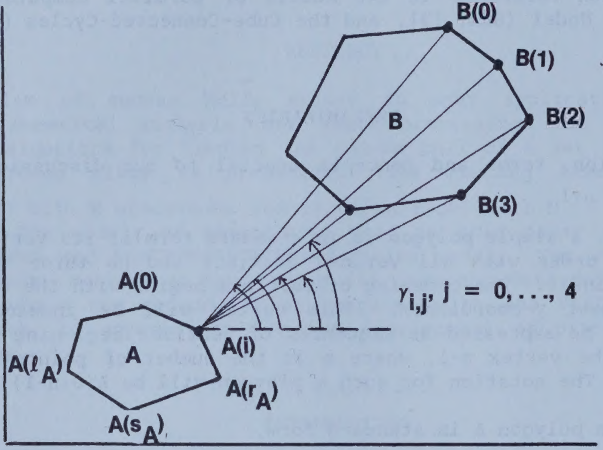


Figure 1. Illustration of $(\gamma_{i,0},\ldots,\gamma_{i,s_B})$

Next, we introduce an important index $j^{(i)}$ which is defined as:
$\min\{j|\ \gamma_{ij} \le \gamma_{ik},\ 0 \le k \le r_B\}$ for i, $0 \le i \le r_A$ and as
$\min\{j|\ \gamma_{ij} \le \gamma_{ik},\ r_B \le k \le s_B\}$ for i, $r_A \le i \le s_A$.

For example in Figure 1, $j^{(2)}=3$. We also define $\bar{j}^{(i)}$ as:
$\max\{j|\ \gamma_{ij} \ge \gamma_{ik},\ s_B \le k \le \ell_B\}$ for i, $s_A \le i \le \ell_A$ and as:
$\max\{j|\ \gamma_{ij} \ge \gamma_{ik},\ \ell_B \le k \le m\}$ for i, $\ell_A \le i \le n$.

With some simple geometric arguments, we can present the following characteristics of the indices $j^{(i)}$ and $j^{(i)}$.

Lemma 1. $\alpha_{i+1,i} < \gamma_{i,j}{}^{(i)} \Rightarrow j^{(i)} \le j^{(i+1)}$, $0 \le i \le s_A$

Lemma 2. $j^{(i)} < j^{(i+1)} \Rightarrow \alpha_{i+1,i} < \gamma_{i,j}{}^{(i)}$, $0 \le i \le s_A$

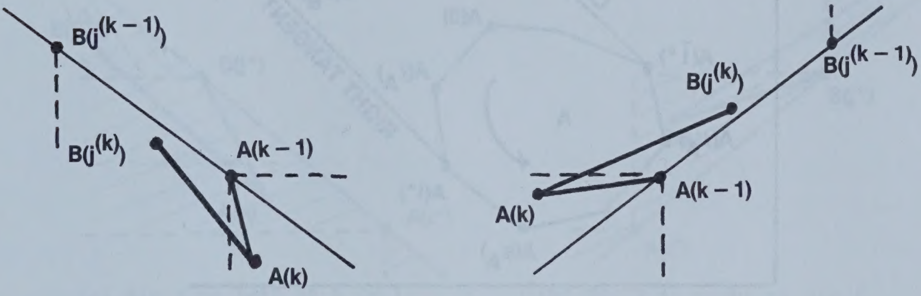
Lemma 3. $\alpha_{i,i+1} < \gamma_{i,j}{}^{(i)} \Rightarrow \bar{j}^{(i)} \ge \bar{j}^{(i+1)}$, $s_A \le i \le n$

Lemma 4. $\bar{j}^{(i)} > \bar{j}^{(i+1)} \Rightarrow \alpha_{i,i+1} < \gamma_{i,j}{}^{(i)}$ $s_A \le i \le n$.

We will use these lemmas to show an important property of the sequence of $j^{(i)}$ $(\bar{j}^{(i)})$.

Theorem 1. In the range $0 \le i \le r_A$, if $j^{(i-1)} < j^{(i)}$ for some i then $j^{(i)} \le j^{(i+1)} \le \ldots \le j^{(r_A)}$. And in the range $r_A \le i \le s_A$, if $j^{(i-1)} < j^{(i)}$ then $j^{(i)} \le j^{(i+1)} \le \ldots \le j^{(s_A)}$.

Proof: We shall show that if $j^{(i-1)} < j^{(i)}$ then $\alpha_{k+1,k} < \gamma_{k,j^{(k)}}$ for $k = i-1, i, \ldots, h$, where h is r_A for $0 \le i \le r_A$ and s_A for $r_A \le i \le s_A$. We prove by induction on k. The basis $\alpha_{i,i-1} < \gamma_{i-1,j^{(i-1)}}$ is true by Lemma 2. In the inductive step, we assume that $\alpha_{k,k-1} < \gamma_{k-1,j^{(k-1)}}$. Then by Lemma 1, $j^{(k-1)} \le j^{(k)}$. Referring to Figure 2, we have $\alpha_{k,k-1} < \gamma_{k,j^{(k)}}$. Due to convexity, $\alpha_{k+1,k} < \alpha_{k,k-1}$. Therefore, we have $\alpha_{k,k+1} < \gamma_{k,j^{(k)}}$. Hence the statement in Lemma 1 completes the proof.



(a) $0 \le k \le r_A$ (b) $r_A \le k \le s_A$

Figure 2. Illustration of a portion of the proof of Theorem 1.

Using an argument similar to the one above, we can establish the following theorem.

Theorem 2. In the range $s_A \le i \le \ell_A$, if $\bar{j}^{(i-1)} > \bar{j}^{(i)}$ for some i then $\bar{j}^{(i)} \ge \bar{j}^{(i+1)} \ldots \ge \bar{j}^{(\ell_A)}$. And in the range $\ell_A \le i \le n$, if $\bar{j}^{(i-1)} > \bar{j}^{(i)}$ then $\bar{j}^{(i)} \ge \bar{j}^{(i+1)} \ge \ldots \ge \bar{j}^{(n)}$.

Theorems 1 and 2 can be interpreted as follows:

Collary 1. $(j^{(0)}, \ldots, j^{(r_A)})$ is a nonincreasing sequence followed by a nondecreasing sequence; so is $(j^{(r_A)}, \ldots, j^{(s_A)})$. $(\bar{j}^{(s_A)}, \ldots, \bar{j}^{(\ell_A)})$ is a nondecreasing sequence followed by a nonincreasing sequence; so is $(\bar{j}^{(\ell_A)}, \ldots, \bar{j}^{(n)})$.

Given two convex polygons $A(0{:}\ n-1)$ and $B(0{:}\ m-1)$, where the y-value of $A(i)$ is smaller than that of $B(j)$, we must find a means of merging them to obtain a single convex hull, denoted as $C(0{:}\ j^*-i^*+\bar{i}^*-\bar{j}^*+m-1)$. First, the two lines of support $(A(\bar{i}^*),B(\bar{j}^*))$ and $(A(i^*),B(j^*))$ common to A and B are traced. These lines are referred to as left and right tangents, respectively. Second, we eliminate the vertices of A and B which become internal to the resulting polygon (refer to Figure 3).

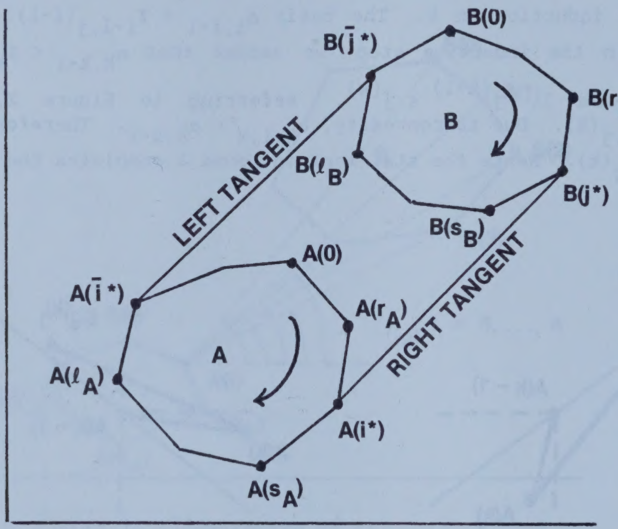


Figure 3. Illustration of the Merging of Two Planar Convex Hulls

We begin by exploring some characteristics of the left and right tangents so that these tangents can be determined efficiently. It is observed that if $B(r_B)$ is to the left of $A(r_A)$, then i^* and j^* are in the ranges $[0,r_A]$ and $[0,r_B]$, respectively; otherwise, i^* and j^* are in the ranges $[r_A,s_A]$ and $[r_B,s_B]$, respectively. It is also observed that if $B(\ell_B)$ is to the left of $A(\ell_A)$, then $\bar{i}^*$ and $\bar{j}^*$ are the intervals $[s_A,\ell_A]$ and $[s_B,\ell_B]$, respectively; otherwise, $\bar{i}^*$ and $\bar{j}^*$ are in the intervals $[\ell_A,n]$ and $[\ell_B,m]$, respectively.

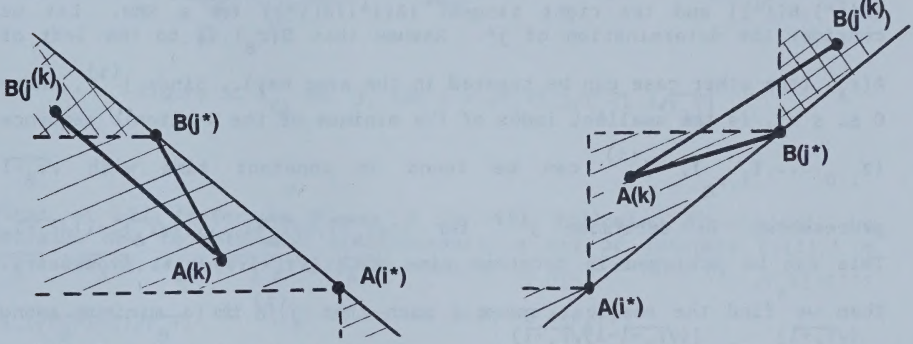
Now the tangents $(A(i^*),B(j^*))$ and $(A(\bar{i}^*),B(\bar{j}^*))$ can be characterized by the following properties.

(1) $j^* = j^{(i^*)}$ and $\bar{j}^* = \bar{j}^{(i^*)}$
(2) $\alpha_{i^*+1,i^*} < \gamma_{i^*j^*} \le \alpha_{i^*,i^*-1}$;
$\alpha_{\bar{i}^*-1,\bar{i}^*} > \gamma_{\bar{i}^*\bar{j}^*} \ge \alpha_{\bar{i}^*,\bar{i}^*+1}$.

The indices j^* and $\bar{j}^*$ have another property which is not so obvious as those above, as expressed by the following lemma:

Lemma 5. $j^* \le j^{(i)}$ for $0 \le i \le r_A$ when $0 \le j^* \le r_B$ and for $r_A \le i \le s_A$ when $r_B \le j^* \le s_B$. $\bar{j}^* \ge \bar{j}^{(i)}$ for $s_A \le i \le \ell_A$ when $s_B \le j^* \le \ell_B$ and for $\ell_A \le i \le n$ when $\ell_B \le j^* \le m$.

Proof: Suppose $j^* < j^{(k)}$ for some k in the appropriate range. Due to property (2) of i^* and j^*, A(k) must be in the hatched region, and due to property (1) and the assumption $j^* > j^{(k)}$, $B(j^{(k)})$ must be in the cross hatched region. We observe from Figure 4 that $\gamma_{k,j^{(k)}} > \gamma_{k,j^*}$ which contradicts the definition of $j^{(k)}$. Therefore, $j^* \le j^{(i)}$ for all i in the specified range. The part on j^* can be proved in a similar manner.



(a) $B(r_B)$ is left of $A(r_A)$ (b) $B(r_B)$ is right of $A(r_A)$

Figure 4. Illustration of the proof of Lemma 5.

The above results suggest that the algorithm for determining the tangents will consist of two steps:

1. find j^* and $\bar{j}^*$;
2. determine i^* and $\bar{i}^*$ which, with j^* and $\bar{j}^*$, satisfy properties (1) and (2).

Having done this, eliminating the vertices of A and B, which become internal to the resulting polygon, is simply a matter of rearranging the vertices of the resulting polygon.

On the Shared Memory Model

Developing algorithms with respect to the SMM, we can ignore the data-movement and concentrate on uncovering the inherent parallelism of the problem. Before describing how to find the common tangents, we have to address two basic problems: finding the minimum of a V-bitonic sequence and finding the maximum of a Λ-bitonic sequence.

Given a V-bitonic (Λ-bitonic) sequence $D(0\colon n-1)$, we want to find the smallest (largest) index k such that $D(k)$ is a minimum (maximum) of the sequence. The index k has the property that $D(k-1) > D(k) \le D(k+1)$ ($D(k-1) \le D(k) > D(k+1)$). Therefore, it is obvious that k can be found in constant time on a SMM with n processors and n memory units.

We are going to solve this problem on a SMM with $\sqrt{n}$ processors and n memory units. We first find the smallest (largest) index i such that $D(i\sqrt{n})$ is a minimum (maximum) of the sequence $(D(\sqrt{n}), D(2\sqrt{n}), \ldots, D((\sqrt{n}-1)\sqrt{n}))$. Note that this sequence is also V-bitonic (Λ-bitonic). It is observed that k must be in the interval $[(i-1)\sqrt{n}+1, (i+1)\sqrt{n}-1]$ which is of length $2\sqrt{n}-1$; $(D(i-1)\sqrt{n}+1), \ldots, D(i\sqrt{n})$ and $D(i\sqrt{n}), \ldots, D(i+1)\sqrt{n}-1))$ are both V-bitonic sequences of length $\sqrt{n}$. Therefore, the index k can be determined in constant time with $\sqrt{n}$ processors.

We now develop an algorithm for an SMM for finding the left tangent $(A(\bar{i}^*), B(\bar{j}^*))$ and the right tangent $(A(i^*), B(j^*))$ for a SMM. Let us consider the determination of j^*. Assume that $B(r_B)$ is to the left of $A(r_A)$ (the other case can be treated in the same way). Since $j^{(i)}$, where $0 \le i \le r_A$, is the smallest index of the minimum of the V-bitonic sequence $(\gamma_{i,0}, \ldots, \gamma_{i,r_B})$, $j^{(i)}$ can be found in constant time with $\sqrt{r_B+1}$ processors. We determine $j^{(i)}$ for $i=\sqrt{r_A+1}, 2\sqrt{r_A+1}, \ldots, (\sqrt{r_A+1}-1)\sqrt{r_A+1}$. This can be achieved in constant time with $(\sqrt{r_A+1}-1)\sqrt{r_B+1}$ processors. Then we find the smallest index i such that $j^{(i)}$ is a minimum among $\{j^{(\sqrt{r_A+1})}, \ldots, j^{((\sqrt{r_A+1}-1)\sqrt{r_A+1})}\}$. This can be done in time $O((\log(\sqrt{r_A+1}-1))^2)$ with $\sqrt{r_A+1}-1$ processors. The index j^* is the smallest in the set $\{j^{(i-\sqrt{r_A+1}+1)}, j^{(i-\sqrt{r_A+1}+2)}, \ldots, j^{(i+\sqrt{r_A+1}-1)}\}$ of size $2\sqrt{r_A+1}-1$. Therefore j^* can be found in $O((\log n)^2)$ time on an SMM with $\sqrt{nm}$ processors. The index $\bar{j}^*$ can be determined in a similar way. The indices i^* and $\bar{i}^*$ are the two i's which satisfy properties (1) and (2) as described. Knowing j^* and $\bar{j}^*$, the indices i^* and $\bar{i}^*$ can be determined in constant time with n processors.

In conclusion, the left and right tangents can be determined in time $O(\log n)$ with at most m+n processors, and it is obvious that the rearrangement can be done in constant time.

On the Cube-Connected Cycles Model

The development of algorithms with respect to the CCC is concerned with not only the data-dependence but also the data-movement. We have to make some modifications to the approach described in last subsection.

Using the facts that j^* is the minimum among the $j^{(i)}$'s and that the sequences of $\gamma_{i,j}$'s are V-bitonic, we can determine j^* as follows. Only the case that $B(r_B)$ is to the left of $A(r_A)$ will be discussed; the other case can be treated in a similar manner.

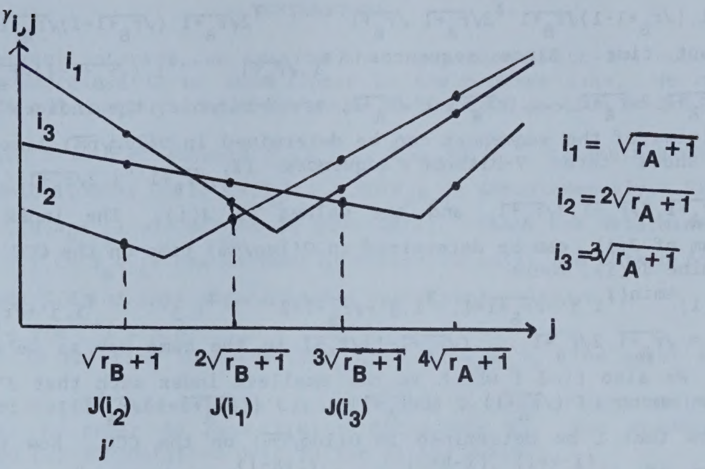


Figure 5. γ_{ij} vs. j, for $i = \sqrt{r_A+1}, 2\sqrt{r_A+1}, 3\sqrt{r_A+1}$

First of all (refer to Figure 5 for the following discussion), we describe how to determine simultaneously a set of integers $\{J(i), i = \sqrt{r_A+1}, 2\sqrt{r_A+1}, \ldots, (\sqrt{r_A+1}-1)\sqrt{r_A+1}\}$, where $\gamma_{i,J(i)} = \min\{\gamma_{i,\sqrt{r_B+1}}, \ldots, \gamma_{i,(\sqrt{r_B+1}-1)\sqrt{r_B+1}}\}$ if $B(r_B)$ is to the left of $A(r_A)$.

To begin with, let us consider two duplicating patterns of a data array D(0: q-1); (i) the first pattern, to be referred to as P1(ℓ) consists in duplicating D ℓ times into {D(0),D(1),...,D(q-1),D(0),...,D(q-1),...} (ii) the second pattern, to be referred to as P2(ℓ), consists in duplicating each element of D ℓ times into {D(0),D(0),..., D(0),D(1),..., D(1),..., D(q-1),..., D(q-1)}. Both patterns have q elements. The first pattern P1(ℓ) can be achieved by copying each element of {D(0),...,D(q-1)} into the module q positions away, then copying each element of {D(0),...,D(q-1),D(0),...,D(q-1)} into the module 2q positions away, and so on. It will take logarithmic steps to achieve the pattern P1(ℓ). We achieve the second pattern P2(ℓ) as follows. We copy D(0),D(1),...,D(q-1) into modules 0,1,2 ,...,(q-1) respectively by a reverse process of the concentration procedure described in [2]. We then perform a selected broadcasting as described in Section 2.2.2 of [2] to achieve pattern P2(ℓ). Both of these operations can be achieved in logarithmic time. Therefore, both patterns can be achieved on a CCC with q processors in O(log(q ℓ)) steps.

We duplicate $\{B(\sqrt{r_B+1}), B(2\sqrt{r_B+1}), \ldots, B((\sqrt{r_B+1}-1)\sqrt{r_B+1})\}$ into pattern $P1(\sqrt{r_A+1}-1)$ and $\{A(\sqrt{r_A+1}), A(2\sqrt{r_A+1}), \ldots, A((\sqrt{r_A+1}-1)\sqrt{r_A+1})\}$ into pattern $P2(\sqrt{r_B+1}-1)$. Now we can compute $\{\gamma_{\sqrt{r_A+1},\sqrt{r_B+1}}, \gamma_{\sqrt{r_A+1},2\sqrt{r_B+1}}, \ldots,$

$\gamma_{\sqrt{r_A+1},(\sqrt{r_B}+1-1)\sqrt{r_B+1}}, \gamma_{2\sqrt{r_A+1},\sqrt{r_B+1}}, \ldots, \gamma_{2\sqrt{r_A+1},(\sqrt{r_B+1}-1)\sqrt{r_A+1}}, \ldots\}$ in constant time. Since sequences $(\gamma_{i,\sqrt{r_B+1}}, \ldots, \gamma_{i,(\sqrt{r_B+1}-1)\sqrt{r_B+1}}\}$, for $i = \sqrt{r_A+1}, 2\sqrt{r_A+1}, \ldots, (\sqrt{r_A+1}-1)\sqrt{r_A+1}$, are V-bitonic, the indices J(i)'s of the minima of the sequences can be determined in $O(\log\sqrt{nm})$ time. Figure 5(a) shows three V-bitonic sequences $(\gamma_{i,\sqrt{r_B+1}}, \gamma_{i,2\sqrt{r_B+1}}, \ldots)$, for $i = \sqrt{r_A+1}, 2\sqrt{r_A+1}, 3\sqrt{r_A+1}$, and the values of J(i). The index j', the minimum of J(i), can be determined in $O(\log\sqrt{nm})$ time on the CCC. We then determine J'(i), where

$\gamma_{i,J'(i)} = \min\{\gamma_{i,j'-\sqrt{r_B+1}+1}, \gamma_{i,j'-\sqrt{r_B+1}+2}, \ldots, \gamma_{i,j'} \ldots, \gamma_{i,j'+\sqrt{r_B+1}-1}\}$

for $i = \sqrt{r_A+1}, 2\sqrt{r_A+1}, \ldots, (\sqrt{r_A+1}-1)\sqrt{r_A+1}$ in the same way as we determine J(i). We also find i which is the smallest index such that J'(i) is a minimum among $\{J'(\sqrt{r_A+1}), J'(2\sqrt{r_A+1}), \ldots, J'((\sqrt{r_A+1}-1)\sqrt{r_A+1})\}$. It is easy to show that i be determined in $O(\log\sqrt{nm})$ on the CCC. Now j^* is the minimum of $\{j^{(i-k+1)}, j^{(i-k+2)}, \ldots, j(^{(i+k-1)}\}$ and can be found in a procedure similar to the one given above.

In an analogous way, we can find $\bar{j}^*$. Knowing j^* and $\bar{j}^*$, i^* and $\bar{i}^*$ can be determined in constant time by finding pairs of (i^*,j^*) and $(\bar{i}^*,\bar{j}^*)$ which satisfy properties (1) and (2) as defined in the previous section. Therefore, the left and right tangents can be determined in time $O(\log(n+m))$ on a CCC with n+m processors. Rearrangement of the vertices to obtain the resulting convex hull involves cyclic forward and backward shift of an array of data which can be implemented on CCC with n+m processors in $O(\log(n+m))$ parallel steps.

CONVEX HULLS ALGORITHM

As a preliminary step, we sort the sets of N points by their y-coordinates in descending order. This can be done in $O((\log N)^2)$ time with N processors in both the SMM and the CCC. The convex hulls algorithm to be presented is a recursive program:

```
function (CH(S))
    begin if N ≤ 2 then return(S)
        return (MERGE(CH(S(N/2: N-1)), CH(S(0: N/2-1))))
    end.
```

The running time T(N) of function CH can be obtained by recurrence relation $T(N) \le T(N/2) + M(N)$, where M(N) is the time in the major step MERGE. It is shown that M(N) is O(logN). Hence, $T(N) = O((\log N)^2)$.

Theorem 2. The convex hull of a set of N points in two dimensions can be determined in time $O((\log N)^2)$ on an SMM with N processors and N memory units, or a CCC with N processors.

EXTENSIONS

For theoretical interest, we examine the improvement with the number of processors increased to be superlinear in the problem size. We modify the "divide and conquer" algorithm given in the last section as follows.

We partition S into N^{α} subsets $S_0, S_1, \ldots, S_{N^{\alpha}-1}$ of $N^{1-\alpha}$ elements each. We then determine convex hulls $CH(S_0), \ldots, CH(S_{N^{\alpha}-1})$ simultaneously. Finally $CH(S_0), \ldots, CH(S_{N^{\alpha}-1})$ are merged to give CH(S). Since the determinations of $CH(S_0), \ldots, CH(S_{N^{\alpha}-1})$ can be done by recursive calls, we obtain for the running time T(N) of this algorithm the recurrence relation

$$T(N) \leq T(N^{1-\alpha}) + M(N),$$

where M(N) is the time to merge $CH(S_0), \ldots, CH(S_{N^{\alpha}-1})$. The major effort is to show that N^{α} convex hulls can be merged in time O(logN) with $2N^{1+\alpha}$ processors, in order to have $T(N) = O(1/\alpha \log N)$. The readers are referred to [2] for detailed description of the merger.

SUMMARY

In this paper, $O((\log N)^2)$ algorithms were presented for determining the convex hull of a set of N points in the plane on both the SMM and the CCC. These results demonstrate that solutions to convex hull problems can be organized to reveal a large amount of parallelism, which can be exploited to substantially reduce computational time.

REFERENCES

[1] Akl, S. G., Toussaint, G. T., "A Fast Convex Hull Algorithm," Information Processing Letters, Vol. 7, No. 5, August 1978, pp. 219-222.

[2] Chow, A. L., "Parallel Algorithms for Geometric Problems," Ph.D. Thesis, Department of Computer Science, University of Illinois, Urbana, Illinois, 1980.

[3] Nassimi, D. and Sahni, S., "Parallel Permutation and Sorting Algorithms and a New Generalized-Connection-Network," Computer Science Department, Tech. Report 79-8, University of Minnesota, Minneapolis, April 1979.

[4] Preparata, F. P. and Hong, S. J., "Convex Hulls of Finite Sets of Points in Two and Three Dimensions," CACM, Vol. 20, No. 2, February 1977, pp. 87-93.

[5] Preparata, F. P., and Vuillemin, J., "The Cube-Connected-Cycles: A Versatile Network of Parallel Computation," Proc. 20th Annual IEEE Symp. on Foundations of Computer Science, October 1979.

[6] Shamos, M. I., Computation Geometry, Department of Computer Science, Yale University, 1977, to be published by Springer-Verlag.

I/O LIMITATIONS ON MULTI-CHIP VLSI SYSTEMS

Extended Abstract

by
Marc Snir
Computer Science Department
New York University
251 Mercer Street, New York, N. Y. 10012

0. ABSTRACT

Decompositions of regular interconnection networks used for multiprocessing are investigated. Lower bounds on the number of components as a function of the number of connections between components are given.

1. INTRODUCTION

The advances in VLSI technology have fostered interest in new computer architectures suited to this technology. In particular, much research has been devoted to the design of multiprocessor systems consisting of large, regular networks connecting identical processors. The same approach has been followed in the design of special-purpose VLSI systems that implement parallel algorithms in hardware. Among the interconnection networks that have been considered are mesh-connected, hexagonally-connected, tree and shuffle exchange networks [2,4]. These are single-stage interconnection networks, with processors at the nodes being directly connected by lines. Alternatively, one may use multistage networks consisting of interconnected switches. A ubiquitous network of this type is the Ω-network [3].

The complexity of VLSI systems is usually measured in area. This measure takes into account the area required for the connecting wires as well as the area required for the computation nodes. For each of the networks mentioned above area-optimal layouts are known. These layouts are obtained under the assumption that the entire network is embedded on one chip. In practice, large multiprocessor systems are built today from tens of thousands of discrete components, and taking into account the human inclination to stretch any technology to its limits, we may safely forecast that large multiprocessor machines will be multi-chip systems in the forseeable future.

When a network is partitioned into several chips we have to take into account the additional restrictions due to pin limitations. The limited bandwidth available for off-chip communication is a major constraint in VLSI environment, which seems to get worse with increased miniaturization. It is therefore important to find strategies for partitioning networks such that the number of connections between distinct components is minimized. We shall study in this paper the existence of such partitions.

2. DEFINITIONS

Let $G=\langle V,E\rangle$ be an (undirected) graph and let $\Pi = \{V_1,\ldots,V_k\}$ be a partition of the nodes of G. We define $b(V_i)$, the boundary of V_i, to be the set of edges connecting nodes from V_i to nodes outside V_i. The size of the partition Π is defined to be the number $s(\Pi) = k$ of subsets it consists of; The weight of the partition Π is defined to be the maximal number $w(\Pi)$ of nodes in a subset V_i of the partition; the girth of the partition Π is defined as $g(\Pi) = \max \#b(V_i)$.

An implementation of a network on several chips defines a partition of the corresponding graph. The chip area restricts the maximal number of nodes that can be implemented on one chip, that is the weight of the partition; the pin limitation restricts the maximum number of lines connecting nodes on one chip with nodes from other chips, that is the girth of the partition. We attempt to minimize the number of distinct components, that is the size of the partition, under these two constraints. We shall investigate the non-trivial cases where the size of the partition is restricted by the partition girth rather then the partition weight.

3. MESH-CONNECTED NETWORKS

A d-dimensional mesh-connected network of size N^d consists of the nodes $\langle n_1,\ldots,n_d\rangle$, $1 \leqslant n_i \leqslant N$. The node $\langle n_1,\ldots,n_d\rangle$ is connected to the nodes $\langle n_1,\ldots,n_j \pm 1,\ldots,n_d\rangle$, $1 \leqslant j \leqslant d$, provided such nodes exist. Each node has at most 2d neighbours. Of particular interest are 2-dimensional grids which have simple planar layouts. With such layout the girth of a set of nodes is related to the perimeter of a surface, whereas the number of nodes in the set is related to its area. We can therefore bound the the maximal size of a set as a function of its girth.

We recall the following facts from planar geometry.

FACT 1. The maximal area of a surface circumscribed by a curve of fixed length is achieved by a circle.

FACT 2. The maximal area of a surface circumscribed by a straight segment and a curve of fixed length is achieved by a half-circle.

FACT 3. The maximal area of a surface circumscribed by two orthogonal segments and a curve of fixed length is achieved by a quadrant.

THEOREM 3.1. Let M be a 2-dimensional mesh-connected network of size N^2. Then for any subset V of nodes from M with $\#V \leqslant N^2/2$ we have $\#V = O(\#b(V)^2)$.

PROOF: Let $b = \#b(V)$ and $v = \#V$. We can assume w.l.g. that $b < N$, the theorem being otherwise trivial . We represent the network by a square grid, with nodes at unit distances. We associate to each node P in V a unit square centered at P. The union of these squares forms a surface S with area v and perimeter b (not counting that part of the perimeter provided by the boundary of the grid - see Fig. 1). Assume that the surface S is the union of connected components S_i, each with area v_i and perimeter b_i, so that $v = \sum v_i$ and $b = \sum b_i$.

As $b < N$ and $\#V \leqslant N^2/2$ no connected component of S can touch more than two of the sides of the surrounding square. It follows therefore that the area of S_i is no larger than the area of a quadrant bounded by an arc of length b, that is $v_i \leqslant {b_i}^2/\pi$. It follows that

$$v = \sum v_i \leqslant \frac{1}{\pi}\sum {b_i}^2 \leqslant \frac{1}{\pi}\left(\sum b_i\right)^2 = b^2/\pi. \qquad \text{QED}$$

CORROLARY 3.2. For any partition Π of a 2-dimensional mesh-connected network of size N^2 we have $s(\Pi) = \Omega\left(N^2/g(\Pi)^2\right)$.

The same argument can be carried through for d-dimensional mesh-connected networks, $d > 2$, by considering regular grids in d-dimensional space. We obtain:

THEOREM 2.3. Let M be a d-dimensional mesh-connected network of size N^d. Then

1. For any set V of nodes from M we have $\#V = O\left(\#b(V)^{d/(d-1)}\right)$.
2. For any partition Π of M we have $s(\Pi) = \Omega\left(N^d/g(\Pi)^{d/(d-1)}\right)$.

Note that the constants implicit in the O notation depend on d. The same lower bound argument is valid for hexagonally connected networks, and for any network that can be embedded in d-dimensional space such that the distance between two nodes is at least 1, and the length of each edge is at most c, for some constant c which is independent of the network size. These lower bounds can be matched in each case by similar upper bounds, by taking the obvious decomposition into subsquares (sub-hypercubes).

4. REGULAR TREES

A regular tree of degree k and depth d consists of the nodes $\langle a_1 \ldots a_j \rangle$, where $0 \leqslant j \leqslant d$ and $1 \leqslant a_i \leqslant k$. The node $\langle a_1 \ldots a_j \rangle$ is connected to the node $\langle a_1 \ldots a_{j-1} \rangle$ (its parent) and to the nodes $\langle a_1 \ldots a_j a_{j+1} \rangle$ (its children), if such exist. The tree contains $(k^{d+1}-1)/(k-1)$ nodes and each node has at most k+1 neighbours.

LEMMA 4.1. A regular tree with N nodes can be partitioned into a partition of weight $\leqslant w$, girth $\leqslant g$ and size $O(N/w)$, for any $g > 2$.

PROOF. Let k be the degree of the tree. We assume w.l.g that $w = (k^{r+1})/(k-1)$. The r bottom levels of the tree can be partitioned into $((k-1)N+1)/((k-1)w+1)$ subtrees of weight w. The remaining $(N-w)/((k-1)w+1)$ nodes can be partitioned into $O(N/kgw)$ subtrees with g-1 outgoing edges at the leaves and one outgoing edge at the root (see Fig. 2). QED

The last result is assymptotically optimal. It implies that girth restrictions do not affect significantly the number of components in a partition of a tree . This is true since most of the nodes of a tree are found at the bottom levels and these levels can be efficiently packed into subtrees with only one outgoing connection. The situation is completely different if we are looking for a partition of the internal nodes of the tree, leaves not included. Such a situation occurs in practice, where the tree network is a connection network with simple switches at the internal nodes, whereas the leaves are more complex processors or memory banks. We have:

LEMMA 4.2. Let V be a set of internal nodes in a tree of degree k. Then $\#b(V) > (k-1)\cdot\#V$.

PROOF. The number e of leaves in a tree of degree k is related to the number n of internal nodes by the relation $e = n(k-1)+1$. Each connected component C of V is a subtree of degree k. If C has c nodes then it has at least $c(k-1)+1$ outgoing edges. The claim now follows. QED

CORROLARY 4.3. Let T be a tree of degree k with N internal nodes. For any partition Π of the internal nodes of T we have $s(\Pi) > N(k-1)/(g(\Pi)-1)$.

Thus, for a partition of the internal nodes of a tree, the limiting factor is the girth rather than the weight.

5. Ω-NETWORKS

Ω-networks have been introduced by Lawrie [3] as interconnection networks for parallel processing. Many other networks considered in the literature are topologically equivalent to Ω-networks. This includes the Flip (Staran) network, the indirect binary cube network, the baseline network, and the SW-banyan network with 2×2 switches [4]. Our results are valid for each of them. They can be also applied to Benes networks which essentially consist of two Ω-networks joined end to end.

We represent for convenience Ω-networks as directed graphs. The network Ω_n has 2^n inputs $I[a_1 \ldots a_n]$, 2^n outputs $O[a_1 \ldots a_n]$, and n columns of 2^{n-1} 2×2 switches $S[a_1 \ldots a_{n-1};j]$, where $a_i=0,1$ and $1 \leqslant j \leqslant n$. The edges of Ω_n connect the input $I[a_1 \ldots a_n]$ to the switch $S[a_1 \ldots a_{n-1};1]$; the switch $S[a_1 \ldots a_{n-1};j]$ to the switches $S[b,a_1 \ldots a_{n-2};j+1]$, $b=0,1$; and the switch $S[a_1 \ldots a_{n-1};n]$ to the outputs $O[a_1 .. a_{n-1},b]$, $b=0,1$. The network Ω_3 is illustrated in Fig. 3. Note that the switches in two adjacent columns of Ω_n are connected by the unshuffle connection.

THEOREM 5.1. Let V be a set of switches in Ω_n with v nodes and girth g. Then $v \leqslant g\ \lg(g)/4$.

PROOF: Since each switch has the same number of incoming edges as of outgoing edges, the number p of edges entering V is equal to the number of edges leaving V. We shall prove by induction on n that $v \leqslant p \lg(p)/2$.

The claim is true for Ω_1, which contains one switch. Assume it is true for Ω_{n-1}. Note that the graph of Ω_n contains two isomorphic copies Ω^0 and Ω^1 of Ω_{n-1}: Ω^ε consists of the subgraph spanned by the inputs $I[\varepsilon,a_2\ldots a_n]$, the switches $S[a_1\ldots a_{j-1},\varepsilon,a_{j+1}\ldots a_{n-1};j]$, for $1 \leqslant j < n$, and the switches $S[a_1\ldots a_n;n]$ (see Fig. 3).

Let V^ε be the subset of nodes of V belonging to Ω^ε, with the exclusion of switches in the last column of Ω_n, let V^2 be the subset of nodes of V in the last column of Ω_n, and let $v^j = \#V^j$. Let p^j be the number of edges entering V^j, and let q^ε, $\varepsilon = 0,1$ be the number of edges connecting nodes from Ω^ε to nodes in V^2. The number of edges leaving V^ε, $\varepsilon = 0,1$, and therefore the number of edges connecting nodes from V^ε to nodes in V^2 is at most p^ε. We have therefore

$$q^\varepsilon \geqslant p^\varepsilon, \text{ for } \varepsilon = 0,1, \text{ and}$$

$$q^0 + q^1 \leqslant p^0 + p^1 + p^2 = p.$$

Each node of V^2 is incident with one edge coming from Ω^0 and one edge coming from Ω^1. It follows that

$$v^2 \leqslant q^\varepsilon, \text{ for } \varepsilon = 0,1.$$

By the inductive assumption, we have

$$v^\varepsilon \leqslant \frac{1}{2} p^\varepsilon \lg p^\varepsilon, \text{ for } \varepsilon = 0,1.$$

Assume w.l.g that $q^0 \leqslant q^1$. Then

$$\begin{aligned} v &= v^0 + v^1 + v^2 \\ &\leqslant \frac{1}{2} p^0 \lg p^0 + \frac{1}{2} p^1 \lg p^1 + q^0 \\ &\leqslant \frac{1}{2} q^0 \lg q^0 + \frac{1}{2} q^1 \lg q^1 + q^0 \\ &\leqslant \frac{1}{2} (q^0 + q^1) \lg (q^0 + q^1) \leqslant \frac{1}{2} p \lg p. \end{aligned}$$

QED

A similar result is proven is [1].

CORROLARY 5.2 For any partition Π of the switches of the network Ω_n we have $s(\Pi) = \Omega\big(n2^n/(g(\Pi) \lg g(\Pi))\big)$.

This lower bound can be matched by a corresponding upper bound. Indeed, let k divide n. We can partition the switches of Ω_n into $(n/k)2^{n-k}$ subsets containing $k2^{k-1}$ switches, each set with 2^k incoming lines and 2^k outgoing lines. Such set will contain 2^{k-1} consecutive switches in one column, and their successors in the next k-1 columns - see Fig. 4.

Note that the connections between components will follow the same pattern as in an Ω-network of degree 2^k: An Ω-network of degree d has

$d \times d$ switches; It has the same formal definition as the usual Ω-network, with addresses being given to base d rather than in binary notation.

In the general case (k does not divide n) it is still possible to partition Ω_n into no more than $\lceil n/k \rceil 2^{n-k}$ subsets, each with girth at most 2^{k+2}.

6. SHUFFLE-EXCHANGE NETWORKS

The shuffle-exchange network SE_n consists of 2^n nodes $\langle a_1 \ldots a_n \rangle$ and $3 \cdot 2^{n-1}$ edges. Node $\langle a_1 \ldots a_n \rangle$ is connected to node $\langle a_2 \ldots a_n, a_1 \rangle$ (shuffle connection), node $\langle a_n, a_1 \ldots a_{n-1} \rangle$ (unshuffle connection), and node $\langle a_1 \ldots \bar{a}_n \rangle$ (exchange connection). We have drawn SE_3 in Fig. 5.

For convenience we define two related networks. The doubly linked shuffle-exchange network DSE_n is obtained from the shuffle-exchange network ES_n by replacing each exchange edge with two edges. The four pins shuffle-exchange network $4SE_n$ is obtained by merging in SE_{n+1} pairs of nodes differing in the last address bit only. It consists of 2^n nodes, each incident to four edges. Node $\langle a_1 \ldots a_n \rangle$ is connected to the nodes $\langle b, a_1 \ldots a_{n-1} \rangle$ and $\langle a_2 \ldots a_n, b \rangle$, for $b = 0,1$. We shall assume that the edges of the four pin shuffle exchange-network are directed in the unshuffle direction, i.e. connect $\langle a_1 \ldots a_n \rangle$ to $\langle b, a_1 \ldots a_{n-1} \rangle$. This last network is closely related to the Ω-network: It is obtained from an Ω-network by coalescing together switches in successive columns that have the same address within the column. We can therefore apply the results obtained for Ω-networks to four pin shuffle-exchange networks.

THEOREM 6.1. Let α, β and γ be constants such that $\alpha > 0$, $\beta < 1$ and $\gamma \leq 1/2e$. Let V be a set of nodes in $4SE_n$ of size $w \leq \gamma 2^n$ and girth g, where $n^\alpha \leq g \leq 2^{\beta n}$. Then $w = O(g \lg g)$.

PROOF: Since each node has two incoming edges and two outgoing edges, there are g/2 edges entering V and g/2 edges leaving V. Let V′ be the set of switches in the Ω-network Ω_n defined by $V' = \{ S[a_1 \ldots a_n ; j] : \langle a_1 \ldots a_n \rangle \in V \}$. The set V′ contains nw nodes and has at most $2w+(n-1)g/2$ incoming edges. It follows, by Th. 5.1, that

$$nw \leq \frac{1}{2}\big(w+(n-1)g/2\big) \lg \big(w+(n-1)g/2\big). \qquad (5.1)$$

Let

$$F(w) = \frac{1}{2}\big(w+(n-1)g/2\big) \lg \big(w+(n-1)g/2\big) - nw. \qquad (5.2)$$

F(w) is decreasing for

$$w < w_{min} = (2^{n-1}/e - (n-1)g/4, \qquad (5.3)$$

increasing afterward, and it has at w_{min} a minimum value of

$$F(w_{min}) = n(n-1)g/2 - 2^n/e\ln 2. \qquad (5.4)$$

From the choice of β and γ it follows that for n large enough $w_{min} > \gamma 2^n$ and $F(w_{min}) < 0$. It follows that $F(w)$ has a unique root w_0 in the interval $[0, w_{min}]$, and that inequality (5.1) is satisfied only if $w \leqslant w_0$. We shall now estimate the root w_0.

Let

$$w_0 = \frac{1}{4}(n-1)g(\frac{n}{\gamma} - 1). \qquad (5.5)$$

Substituting for w_0 in the equation $F(w_0) = 0$, we obtain

$$\gamma - \lg \gamma = n - \lg \left(n(n-1)g/2\right). \qquad (5.6)$$

Let $C = n - \lg \left(n(n-1)g/2\right)$. For n large enough $C > 1$.

It is easy to see now that the iteration

$$y_0 = C,$$
$$y_{n+1} = C + \lg y_n,$$

produces a sequence that converges to the root γ of the equation (5.6). Also, since $C > 1$, it follows by induction that $y_n \geqslant C$, and therefore, that $\gamma \geqslant C$. Substituting back, we obtain that

$$\begin{aligned} w_0 &\leqslant \frac{1}{4}(n-1)g(n/C-1) \\ &= \frac{(n-1)(n-C)g}{4C} \\ &= \frac{(n-1)(\lg n + \lg(n-1) + \lg g - 1)}{4(n - \lg n - \lg(n-1) - \lg g + 1)} \\ &\leqslant \frac{(n-1)(\lg n + \lg(n-1) + \lg g - 1)}{4((1-\beta)n - \lg n - \lg(n-1) + 1)} \\ &= O(g \lg g)) \end{aligned}$$

QED

CORROLARY 6.2. The claim of theorem 6.1 is valid for subsets of (i) the doubly linked shuffle-exchange network, and (ii) of the shuffle exchange network.

PROOF: (i) Let V be a set of nodes in DSE_n. Assume that the node $\langle a_1 \ldots a_n \rangle$ is in V whereas the node $\langle a_1 \ldots \bar{a}_n \rangle$ is not in V. Then the set $V' = V + \{\langle a_1 \ldots \bar{a}_n \rangle\}$ contains one more node and no more outgoing edges than V. It follows that we can assume w.l.g. that $\langle a_1 \ldots a_{n-1}, 0 \rangle \in V$ iff $\langle a_1 \ldots a_{n-1}, 1 \rangle \in V$. The claim now follows immediatly from Th. 6.1.

(ii) If V is a set of weight w and girth g in SE_n then the corresponding set of nodes in DSE_n has weight w and girth $\leqslant 2g$. The claim follows therefore from (i). QED

As any fixed permutation can be performed on the shuffle exchange network SE_n in $O(n)$ routing steps [5], we must have for any set V of nodes in SE_n $n \# g(V) = \Omega(\#V, 2^n - \#V)$. This yields a new proof of the inequality in Th. 6.1, for large g and v. Some of the restrictions in this theorem can be relaxed, yielding

CORROLARY 6.3. Let α and β be constants such that $\alpha > 0$ and $\beta < 1$. Let V be a set of nodes in SE_n (DSE_n, $4SE_n$) of size $w \leqslant \beta 2^n$ and girth $g \geqslant n^{\alpha}$. Then $w = O(g \lg g)$.

CORROLARY 6.4. Let Π be a partition of the shuffle-exchange network SE_n (DSE_n, $4SE_n$) of weight w and girth g, where $w \leqslant \beta 2^n$ and $n^{\alpha} < g$, for $\alpha > 0$, and $\beta < 1$. Then $s(\Pi) = \Omega(2^n/g \lg g)$.

The lower bounds given in Cor. 6.3 are the best possible of this form: For any $k < n$ it is possible to build in SE_n a set of nodes of size $\Theta(2^k)$ and girth $\Theta(k2^k)$. Nevertheless, it is not obvious that the lower bound of Cor. 6.4 can be matched by a corresponding upper bound.

7. CONCLUSION

Our results show that in order to take full advantage of the increasing gate count G of chips the pin count P must be increased at least as $\sqrt{G}$ for architectures using 2D mesh-connected networks, as G for tree-like communication networks, as G/lg G for Ω-networks and shuffle exchange networks. The past trend seems to indicate a relation of the form $P = \alpha G^{\beta}$, with $\beta < 1/2$. If this can be extrapolated, then the performance of any of the above structures will be restricted by the pin count of its components rather than by their gate count.

ACKNOWLEDGMENTS

I wish to thank Larry Rudolph and Chee Yap for their helpful comments.

REFERENCES

1. Hong, J. W. and Kung, H. T. I/O complexity: the red-blue pebble game. ACM 13th Annual Symp. on Theory of Computing (1981), 326-333.
2. Kung, H. T. The structure of parallel algorithms. Advances in Computers 19 (1980), 65-112.
3. Lawrie, D. H. Access and alignment of data in an array processor. IEEE Trans. Comput. C-24 (1975), 1145-1155.
4. Siegel, H. J. Interconnection networks for SIMD machines. Computers 6 (1979), 57-65.
5. Waksman, A. A permutation network. J. ACM 15 (1968), 159-163.

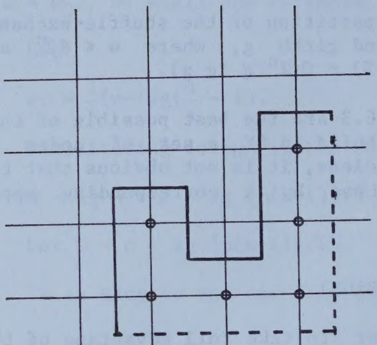


Figure 1
Subset of 2D grid

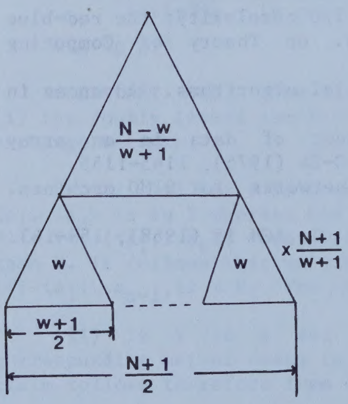

$\frac{N-w}{w+1}$
w
w
$x\ \frac{N+1}{w+1}$
$\frac{w+1}{2}$
$\frac{N+1}{2}$

Figure 2
Partition of a binary tree

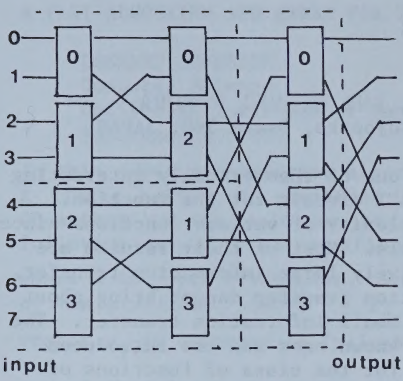

0
1
2
3
4
5
6
7
0
1
2
3
0
2
1
3
0
1
2
3
input
output

Figure 3
Ω_3

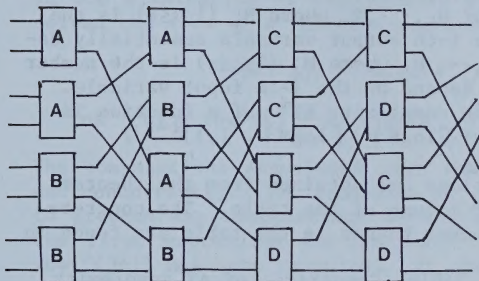

A
A
B
B
A
B
A
B
C
C
D
D
C
D
C
D

Figure 4
Partition of Ω_4

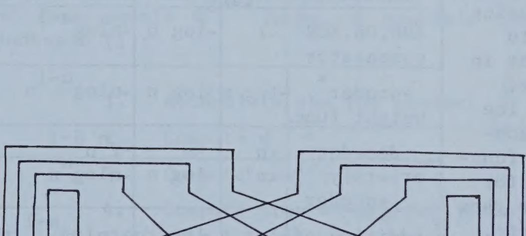


Figure 5
SE_3

AREA- TIME BOUNDS FOR VLSI CIRCUITS

KOUICHI WADA, KENICHI HAGIHARA, AND NOBUKI TOKURA
Faculty of Engin. Sci., Osaka Univ., Toyonaka, Osaka 560, JAPAN

Area-time complexity of a function is one of theoretically interesting measure to judge the design quality of a VLSI circuit for the function. Many reseachers have studied area-time complexity of various functions since Thompson and Brent-Kung presented VLSI models. Most of their results are concerned with the functions with comparatively large information transfer. However, their techniques based on information transfer cannot bring about non-trivial results for the functions with small information transfer. The results in this paper are complementary to known ones and the introduced technique to derive lower bounds is useful for the class of functions of rather simple structure. Good examples of them are decode-function, symmetric-one, etc. It should be noted that the authors assume the bounded fanout condition in addition to the hypotheses of Brent-Kung's model in deriving lower bounds. This is only a partial summary of our results and reader could refer to authors' reports [1] - [3] for further details.

Let $F(n,m,N,M)$ denote the family of functions with the following properties: (1) the number of input variables is n, (2) the number of output variables is m, (3) N is the maximum of $N_1,\cdots,N_m$ where N_i $(1\leq i\leq m)$ is the number of input valiables on which the i-th output variable essentially depends, and (4) M is the maximum of $M_1,\cdots,M_n$ where M_j $(1\leq j\leq n)$ is the number of output variables which essentially depend on the j-th input variable.

[THEOREM] The lower bound of area-time complexity AT^{α} of a function in $F(n,m,N,M)$ is $AT^{\alpha} = \Omega\ (\ \max(n,m)[\max(\lceil \log N\rceil\ ,\ \lceil \log M\rceil\ ,\ 1,)]^{\alpha-1}\ )$ where $\lceil x\rceil$ denotes the ceiling of x. □

The lower bounds of various functions are obtained from the theorem, and some of them are summarized in AT^{α} column of the table. The concrete layouts of circuits that achieve the upper bounds in the table are found in [1] - [3]. An area-time efficient realization of combinational unilateral one-dimensional iterative circuits has been obtained by using Unger's method. This is used to give constructions for functions in the first two rows in the table.

There are no gaps between the lower and upper bounds within constant factor about the starred function in the table. It is left for future study to investigate the gaps about the symmetric function, etc.

Table. Comparison of AT^{α} complexity

function	$A_{[3]}$	T	AT^{α} $(\alpha \geqslant 1)$
AND,OR,XOR* comparator	~ 1	$\sim \log n$	$\sim n\log^{\alpha-1} n$
encoder* weight func.	$\sim \log n$	$\sim \log n$	$\sim n\log^{\alpha-1} n$
decoder*	$\sim n$	$\sim n$	$\sim 2^n n^{\alpha-1}$
priority encoder*	$\sim n^{\dagger}$	$\sim \log n$	$\sim n\log^{\alpha} n$
addition of n t-bit int.	$\sim \log n$	$\sim \log tn$	$\geqslant tn\log^{\alpha-1} tn$ $\leqslant G(t,n,\alpha)^{\dagger\dagger}$
symmetric func. $(S^n_{a_1 \ldots a_k})$	——	$\sim \log n$	$\sim n\log^{\alpha-1} n^{\dagger\dagger\dagger}$ $\leqslant n\log^{\alpha} n^{\dagger\dagger\dagger\dagger}$
threshold func.	——	$\sim \log n$	$\geqslant n\log^{\alpha-1} n$ $\leqslant n^3\log^{\alpha+1} n$

† simultaneous inputs condition.
†† $G(t,n,\alpha)=\{tn+(t\log n)^2\}\log^{\alpha} tn$.
††† $k=O(n/\log^3 n)$, †††† any k.
$\geqslant h(n)=\Omega(h(n))$, $\leqslant h(n)=O(h(n))$,
$\sim h(n)=\Omega(h(n))\&O(h(n))$.

[References] [1] K.Wada, K.Hagihara and N.Tokura: "Area-time lower bounds for VLSI circuits," Prog. Lang. Group Memo No.81-01, Dept. of Inf. & Comput. Sci., Osaka Univ. (April 1981).

[2] - : "The area-time complexity of several functions on a VLSI model," -,No.81-02, -,(Sept. 1981).

[3] - : "The area complexity on a VLSI model," -,No.81-03,-, (Sept. 1981) (in Japanese).

A VLSI ALGORITHM AND ARRAY FOR THE QR-METHOD

LENNART JOHNSSON
Computer Science
California Institute of Technology
Pasadena CA 91125

In this paper a concurrent algorithm for the QR-method is briefly described. A linear system of equations is often written as Ax = y. Applying the QR-method to the solution of this system implies as a first stage the following set of transformations: $P_{n-1}\dots P_1\ Ax = P_{n-1}\dots P_1 y$, or $Rx = Q^{-1}y$, where R is upper triangular. We will, in this short note, focus on this set of computations. Let $A' = P_{j-1}\ P_{j-2}\dots\ P_1 A$, where $P_j = I - 2w_j w_j^*$. The vector w_j is computed as follows

$$u_k = 0,\ k < j, \qquad u_j = a'_{jj} + \mathrm{sign}(a'_{jj})d \qquad u_k = a'_{kj},\ k > j$$

$$d = \mathrm{sqrt}\Big(\sum_{k=j}^{n} a'^{2}_{kj}\Big)$$

$$w_j = u/\mathrm{sqrt}(2d^2 + 2|a'_{jj}|d)$$

In the elimination of the nonzero elements below the diagonal in the i:th column the rows of A′ are changed from row′ to row′′ as follows:

$$\mathrm{row}''(j) = -\mathrm{sign}(a'_{jj})\Big(\sum_{k=j}^{n} a'_{kj}\mathrm{row}'(k)\Big)/d$$

$$\mathrm{row}''(k) = \mathrm{row}'(k) - a'_{kj}\mathrm{sign}(a'_{jj})(\mathrm{row}'(j)-\mathrm{row}''(j))/(d+|a'_{jj}|) \quad k > j$$

The computations associated with the different row elements can be carried out concurrently. The summation of row products in the first expression has to preceed the computations in the second expression. The computation of the first expression for the next column, j+1, can start the cycle after row′′(j+1) has been computed. It should be noticed that the computation of row′′(j) cannot be completed until all the preceeding rows row′(k), k=j, ..., n, has been computed. The first element in the linear combination of rows equals d^2. Hence, a possible computation scheme is as follows, Johnsson [1]:

1. Accumulate the row product $\sum_{k=j}^{n} a'_{kj}\mathrm{row}'(k)$
2. Compute d
3. Compute row′′(j)
4. Compute prow = $-\mathrm{sign}(a'_{jj})(\mathrm{row}'(j)-\mathrm{row}''(j))/(d + |a'_{jj}|)$
5. Compute row′′(k) = $\mathrm{row}'(k) + a'_{kj}\mathrm{prow}$

An array performing the computations above is shown in Figure 1. For a band matrix equation with a bandwidth of r+s+1 (r elements below the diagonal and s above) r+s+1 cells are required for the matrix part. The operations on the right hand side of the equations can be handled by one cell or each column of the right hand side. The back substitution part is identical to the computations for Gaussian elimination, S. Y. Kung [3] and Johnsson [2]. The time for computing the matrix R by the QR-method can be made to be (r+3+a)n in a fully pipelined array with no broadcasting. In the past expression a is the number of multiply add times required to compute the square root of d and n the dimension of the matrix.

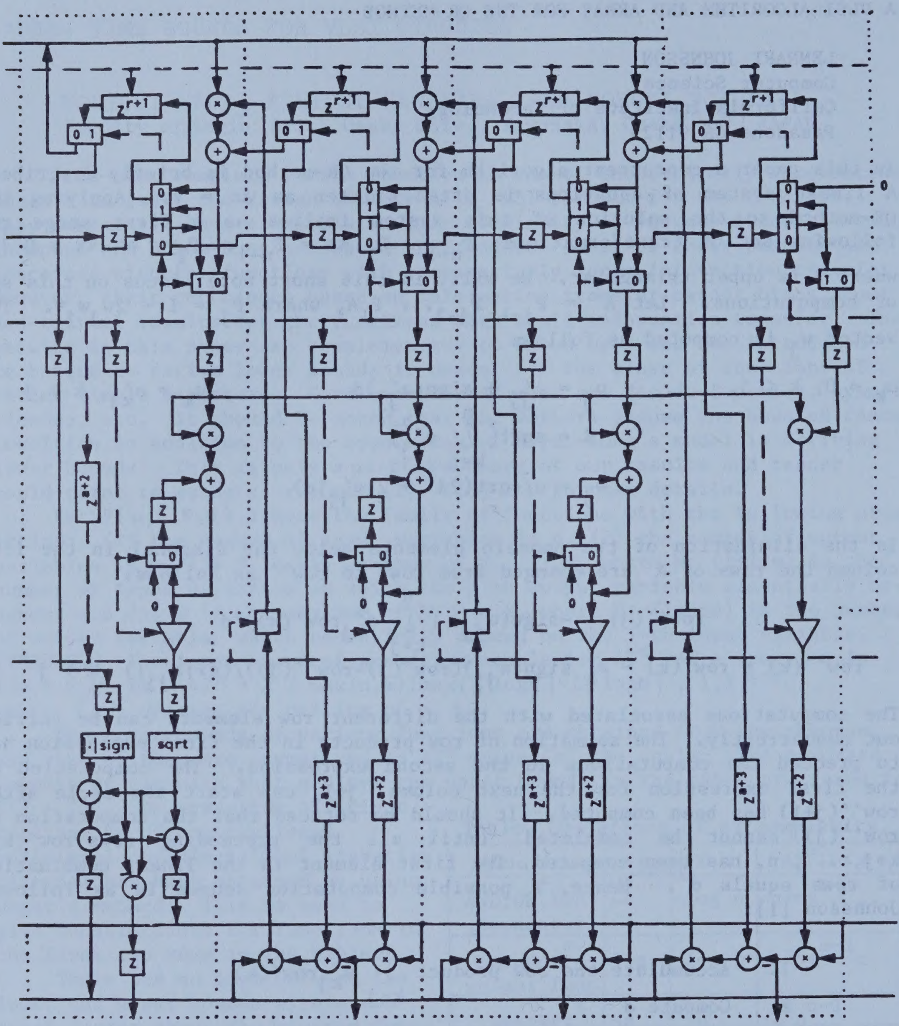


Figure 1. An array for the QR-Method, bandwidth = 4

REFERENCES

[1] Johnsson, S. Lennart, "A Computational Array for the QR-Method," Memo #4533, Computer Science, California Institute of Technology June 1981

[2] Johnsson, S. Lennart, "Computational Arrays for Band Matrix Equations," Memo #4287, Computer Science, California Institute of Technology, May 1981

[3] Kung, S. Y., "VLSI Matrix Computation Array Processor," The MIT Conference on Advanced Research in Integrated Circuits, February 1980

AVERAGE ERROR PROBABILITY FOR DS-SSMA COMMUNICATIONS: THE GRAM-CHARLIER EXPANSION APPROACH

KOU-TAN WU
Computer Information and Control Engineering Program
University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

The average bit error probability for a binary asynchronous direct sequence spread spectrum multiple access system with or without white Gaussian noise is evaluated using the Gram-Charlier expansion approach. This method provides a good approximation to the average error probability for a larger number of users than previous techniques. However, if the number of users is few or the signal to Gaussian noise ratio is high, this method may not offer good results. Fortunately, previous methods can be used in this situation. Numerical results are presented for a set of linear m-sequences and a Gold code.

INTRODUCTION

Asynchronous direct sequence spread spectrum multiple access (DS-SSMA) communication systems [1] are a common form of spread spectrum multiple access modulation. In the transmitter of each user, the carrier is phase-modulated by the digital data and a high rate code sequence. Distinct code sequences are assigned to all users, but since all users share the same frequency band, some mutual interference (multiple access noise) is unavoidable. Therefore the design of this system requires a careful assignment of code sequences to users, so that the multiple access noise is minimized and good performance is achieved; i.e., high signal to noise ratio or low average error probability. A formula for the average signal to noise ratio was given in [1].

Because of the complicated nature of the multiple access noise in such system, it is considered to be more difficult to evaluate the average error probability. However, in [2] upper and lower bounds to the average error probability were evaluated by the moment space bounding technique. In [3] and [4] upper and lower bounds were evaluated by the convexity of the Q function, Jensen's inequality and the trapezoidal integration rule. In [5] the approximation of the average error probability were evaluated by the series expansion method and the Gauss quadrature rule. In [6] the characteristic function method was applied to evaluate the approximation of the average error probability. In [2]-[4] the complexity of each method increases rapidly with the number of users. In [5] both methods require the evaluation of a number of the moments of the multiple access noise, with complexity increasing with the product of the number of users and the square of the codelength plus the square of the number of moments. In addition the number of moments required increases with the number of users and decreases with the signal to Gaussian noise ratio, hence both methods are limited by the number of users, with the Gauss quadrature rule allowing more users than the series expansion method. In [6] the complexity increases linearly with the number of users and the codelength, yet requires a large number of function evaluations. If the signal to Gaussian noise ratio is high, the results may need to be refined by some form of the series expansion method to obtain the desired accuracy. It is not clear that this method will work for a very large number of users.

Our goal is to find another method such that a larger number of users can be allowed in the analysis. The Gram-Charlier expansion [7] is the method we use. It has been applied to the evaluation of error probability for digital communication systems with intersymbol interference and co-channel interference. This method requires the moments of the multiple access noise.

DS-SSMA SYSTEM

We now briefly describe the model for an asynchronous DS-SSMA system. Assume that there are K users transmitting binary data in a common band with center frequency f_c. Let P be the power available to each user's transmitter. The transmitted signal from the k-th user is

$$s_k(t) = \sqrt{2P}\, b_k(t)\, \tilde{a}_k(t) \cos(2\pi f_c t + \theta_k) \qquad (1)$$

where $b_k(t)$ is called the data signal and is a rectangular waveform which equals $b_{k,\ell}$ in the time interval $[\ell T, (\ell+1)T]$ and $b_{k,\ell}$ denotes the ℓ-th bit from the k-th user's source and is assumed to be $\pm$ 1 with equal probability and independent of all other $b_{i,j}$'s, where $\tilde{a}_k(t)$ is a rectangular waveform consisting of periodic repetitions of a code waveform $a_k(t)$ which has duration T seconds and is made up of N rectangular chips, each with duration $T_c = T/N$ seconds and amplitude $\pm$ 1, and where θ_k is the carrier phase. The code waveform $a_k(t)$ equals a_k^j in the time interval $[jT_c, (j+1)T_c]$, for some $\pm$ 1 valued sequence $a_k^0, a_k^1, \ldots, a_k^{N-1}$ called the codeword. The bandwidth of the signal is proportional to $T_c^{-1} = N/T$, so N can be called the bandwidth expansion factor.

The receiver for the k-th user receives the signal

$$r_k(t) = \sqrt{2P} \sum_{k=1}^{K} a_k(t-\tau_k) b_k(t-\tau_k) \cos(2\pi f_c t + \phi_k) + n_k(t) \qquad (2)$$

where $n_k(t)$ is white Gaussian noise that is independent of all other r.v.'s and has two-sided power spectral density $N_o/2$, and where the time delay τ_k

and phase $\phi_k = \theta_k - 2\pi f_c \tau_k$ model the asynchronism between users. We assume that τ_k is uniformly distributed on $[0,T]$, and that ϕ_k is uniformly distributed on $[0,2\pi]$, and that τ_k and ϕ_k are independent of each other and all other r.v.'s in this model.

The receiver for the k-th user is assumed to coherently correlate $r_k(t)$ with $a_k(t-\tau_k)\cos(2\pi f_c t + \phi_k)$ over the interval $[\tau_k, T + \tau_k]$ and make decision about $b_{k,0}$ based on the sign of this correlation. We will concentrate entirely on receiver number one and, without loss of generality, we will assume $\tau_1 = \phi_1 = 0$. Assuming $f_c \gg T_c^{-1}$, the output of this correlation is

$$Z_1 = \sqrt{P/2}\, T b_{1,0} + \sqrt{P/2}\, u + n \tag{3}$$

where the first term is due to the desired signal, the second term is due to interference from other users which will be called the multiple access noise, and n is due to the white Gaussian noise. In particular n is a zero mean Gaussian r.v. with variance $N_o T/4$ and is independent of all other r.v.'s. With slight abuse of terminology, we call either $\sqrt{P/2}\, u$ or u alone the multiple access noise, which is

$$u = \sum_{k=2}^{K} (b_{k,-1} R_{k,1}(\tau_k) + b_{k,0} R_{k,1}(\tau_k)) \cos\phi_k = \sum_{k=2}^{K} w_k \cos\phi_k \tag{4}$$

where

$$R_{k,1}(\tau) = \int_0^{\tau} a_k(t-\tau) a_1(t)\,dt; \quad R_{k,1}(\tau) = \int_{\tau}^{T} a_k(t-\tau) a_1(t)\,dt \tag{5}$$

In this paper system performance is determined by the average error probability P(E) for the first user. From now on, we assume $b_{1,0} = 1$ and compute $P(Z_1 < 0)$. By a symmetry argument, this equals the average error probability P(E).

It is shown in [5] that the moments of the multiple access noise u can be evaluated straightforwardly. We will use these moments to evaluate the average error probability via the Gram-Charlier expansion approach.

AVERAGE ERROR PROBABILITY
BY GRAM-CHARLIER EXPANSION

The Gram-Charlier expansion makes use of the expansion of an unknown pdf in terms of the derivatives of a known pdf. The series representation of error probability is obtained from term by term integration.

Let us define a r.v. x that is related to the sum of multiple access noise and Gaussian r.v. n, as

$$x = x_o\left(\frac{u}{T} + \sqrt{\frac{2}{P}}\, \frac{n}{T} \right); \quad x_o = \left(\frac{E[u^2]}{T^2} + \frac{N_o}{2PT} \right)^{-\frac{1}{2}} \tag{6}$$

where $E_b = PT$ is the signal energy per bit, and where x_o is the ratio of the signal to the sum of multiple access noise and Gaussian r.v. n, and is used such that the r.v. x has unit variance. The expression for the average error probability is thus

$$P(E) = P(x< x_o) = \int_{-\infty}^{-x_o} f_x(x)dx \tag{7}$$

where $f_x(x)$ is the pdf of the r.v. x. A direct numerical computation of the pdf of x requires K convolutions for K users and is in general quite complicated. If there are a sufficient number of users, i.e., K is large, it is suggested by the central limit theorem that the pdf of x tends to the normal density. It is therefore natural to choose the normal density and its derivatives to approximate the pdf of x; this leads to the Gram-Charlier expansion. In particular the pdf of x can be expressed as [7],[8]

$$f_x(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2}) \sum_{n=0}^{\infty} c_n He_n(x) \tag{8}$$

where $He_n(x)$ is called the Hermite polynomial of degree n and is defined by

$$He_n(x) = (-1)^n \exp(-\frac{x^2}{2}) \frac{d^n}{dx^n} \exp(-\frac{x^2}{2}) \tag{9}$$

The coefficients c_n are obtained using the orthogonality relations of the Hermite polynomials; they are expressed by

$$c_n = \frac{\int_{-\infty}^{\infty} f_x(x)He_n(x)dx}{\int_{-\infty}^{\infty} He_n^2(x) \quad 1/\sqrt{2\pi}\ \exp(-x^2/2)dx} = \frac{E_n}{n!} \tag{10}$$

where

$$E_n = \int_{-\infty}^{\infty} f_x(x)\ He_n(x)dx = E[\ He_n(x)] \tag{11}$$

The coefficient E_n is a generalized moment and can be related to the moments of x. Since the pdf of x is even and since $He_n(x)$ is even or odd whenever the degree is even or odd, E_n equals zero when n is odd. The even order Hermite polynomial and the generalized moment E_{2n} can be expressed by

$$He_{2n}(x) = \sum_{k=0}^{n} h_{2n,2k} x^{2k}; \quad E_{2n} = \sum_{k=0}^{n} h_{2n,2k}\ E[x^{2k}] \tag{12}$$

where the $h_{2n,2k}$ are the coefficients of the orthogonal polynomial $He_{2n}(x)$, and can be calculated by the recursive relation [8]

$$h_{2n,2k} = -\frac{n(2n-1)}{n-k} h_{2n-2,\ 2k}; \quad k=0,\ 2,\ 4,\ \ldots,\ n-2$$
$$h_{2n,2n} = 1, \quad \text{for all } n \tag{13}$$

Now the coefficients E_{2n} are related to the even moments of the r.v.x, and the moments can be evaluated as follows

$$E[x^{2k}] = x_o^{-k} \sum_{i=0}^{k} \binom{2k}{2i} \frac{E[u^{2i}]}{T^{2i}} \frac{(2k-2i)!}{2^{k-i}(k-i)!} \left(\frac{N_o}{2E_b}\right)^{k-i} \tag{14}$$

The Gram-Charlier expansion of $f_x(x)$ can now be expressed as

$$f_x(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2}) \sum_{n=0}^{\infty} \frac{E_{2n}}{(2n)!} He_{2n}(x) \qquad (15)$$

The series representation of the average error probability follows by substituting eq. (15) into eq. (7) and using term by term integration. This yields

$$P(E)= Q(x_o) - \sum_{n=1}^{\infty} \frac{E_{2n}\exp(-x_o^2/2)}{(2n)!\sqrt{2\pi}} He_{2n-1}(-x_o) \qquad (16)$$

The first term is the error probability that would be due to additive Gaussian noise with the power equal to that of the multiple access noise plus that of the actual white Gaussian noise. The rest of the series is related to the parameter x_o, the Hermite polynomials and the generalized moments E_{2n}. A finite number of terms can be summed to approximate the average error probability.

This method works best when the number of users is large and the signal to Gaussian noise ratio is low, for then it requires very few terms (2 or 3) to approximate the average error probability. If the number of users is very few and the signal to Gaussian noise ratio is very high, this method requires many terms and so does not work well. However, there are other methods [5] which can be applied in these situations.

IRREDUCIBLE ERROR PROBABILITY

Suppose that white Gaussian noise is not present. The average error probability due to multiple access noise alone is always less than the average error probability for nonzero white Gaussian noise, and is therefore called the irreducible (average) error probability [9]. The irreducible error probability can be thought of as an asymptotic error probability for DS-SSMA system in white Gaussian noise as E_b/N_o goes to infinity.

The output from the correlation receiver for user number one in this case is given by

$$Z_1 = \sqrt{P/2}\ Tb_{1,0} + \sqrt{P/2}\ u \qquad (17)$$

Assuming $b_{1,0}=1$, and using a symmetry argument, we find that $P(Z_1 < 0)$ equals the irreducible error probability, which is denoted $P_{irr}(E)$.

$$P_{irr}(E)= P(\sqrt{P/2}\ T + \sqrt{P/2}\ u < 0)= P(\frac{u}{U} < -\frac{T}{U})= \int_{-1}^{-T/U} f_y(y)dy \qquad (18)$$

where $U=\max|u|<\infty$, where $y = u/U$ and where we have used the fact that the pdf of u is even. It is clear that the pdf of y is even, the odd moments of y are zero, and the even moments of y are related to the even moments of u by

$$E[y^{2i}] = U^{-2i}E[u^{2i}]; \quad i=0, 1, \ldots \qquad (19)$$

Since the r.v. y lies in a finite interval, the pdf of y is conventionally expanded by a family of polynomials called Jacobi polynomials which are orthogonal with respect to the beta density [8],[10], namely,

$$P_{beta}(y) = \begin{cases} c_o(1-y)^{\alpha}(1+y)^{\beta}, & \alpha,\beta > -1, \quad -1 < y < 1 \\ 0 & , \quad \text{elsewhere} \end{cases} \tag{20}$$

Because the pdf of y is even , we choose $\beta=\alpha$, so that the beta density is even. The parameter α is chosen so that the variance of the beta density equals the variance of the r.v. y [10]. Our choice of α is thus related to the second moment of y via

$$\alpha = (1-3E[y^2])/(2E[y^2]) \tag{21}$$

We now summarize our results for $P_{irr}(E)$. These derivation is similar to that of the previous section. The pdf of y can be expressed by

$$f_y(y) = (1-y^2)^{\alpha} \sum_{k=0}^{\infty} \frac{E_{2k}}{h_{2k}(\alpha)} P_{2k}^{(\alpha,\alpha)}(y); \; -1 < y < 1 \tag{22}$$

where $P_{2k}^{(\alpha,\alpha)}(y)$ are called the Jacobi polynomials [12], and where

$$E_{2k} = \int_{-1}^{1} f_y(y)\, P_{2k}^{(\alpha,\alpha)}(y)dy = E[\, P_{2k}^{(\alpha,\alpha)}(y)] \tag{23}$$

$$h_{2k}(\alpha) = \frac{2^{2\alpha+1}\,\Gamma^2(2k+\alpha+1)}{(4k+2\alpha+1)(2k)!\,\Gamma(2k+2\alpha+1)} \tag{24}$$

$$\Gamma(z) = \int_0^{\infty} \exp(-t)\; t^{z-1}dt \tag{25}$$

The coefficients E_{2k} are generalized moments and can be related to the moments of y in the same manner in the previous section. The series representation of $P_{irr}(E)$ can be obtained by substitution and term by term integration. This gives

$$P_{irr}(E) = \frac{1}{h_0(\alpha)} \int_{-1}^{-T/U} (1-\, y^2\,)^{\alpha} dy \; - \sum_{k=1}^{\infty} \frac{E_{2k}}{4k\; h_{2k}(\alpha)} \left(1-\frac{T^2}{U^2}\right)^{\alpha+1} P_{2k-1}^{(\alpha+1,\alpha+1)}\left(-\frac{T}{U}\right) \tag{26}$$

A finite number of terms can be summed to approximate $P_{irr}(E)$. As the number of users increases, the pdf of the multiple access noise can be better approximated by the beta density and the first few terms. Therefore a few terms will suffice to evaluate the irreducible error probability.

NUMERICAL RESULTS

As the first example, a set of 31 codewords of length 127 were chosen arbitrarily from the AO/LSE and LSE/AO sequences described in [11] and assigned to 31 users. The average error probability and the irreducible error probability were calculated for user number one as functions of the number of transmitting users K, assuming codewords 1, 2,..., K were used. In Fig. 1 we have plotted the results of these calculations using the Gram-Charlier expansion. In Table 1, we have listed the number of terms required to maintain 3 significant digits with the Gram-Charlier expansion. It shows that 2 or 3 terms are enough when the number of users is large and the signal to Gaussian noise ratio is low. In Fig. 1, to the left of the left dashed line shows the region where the series expansion method [5] works well, and to the left of the right dashed line shows the region where the Gauss quadrature rule [5] also works well, but is more complicated than the Gram-Charlier expansion method. The region where the Gram-Charlier expansion does not work does not appear in Fig. 1.

In the second example, a set of 45 Gold sequences of length 127 were generated and the same calculations were performed. The results are shown in Fig. 2.

In both examples, the average error probability based on the Gaussian assumption [1] is evaluated for comparison. It yields values that are smaller but are close to the average error probability when the number of users is large and the signal to Gaussian noise ratio is low.

The Gram-Charlier expansion dose not work well if the number of users is very small and the signal to Gaussian noise ratio is high, but in this situation the series expansion method or the Gauss quadrature rule will work [5]. If we combine these methods, the average error probability can be efficiently evaluated in all cases.

ACKNOWLEDGMENT

The author would like to thank Professor David Neuhoff for his review and comments, which improved the presentation of this paper.

REFERENCES

[1] M. B. Pursley,"Performance evaluation for phase coded SSMA communication - Part I: System analysis", IEEE Trans., vol. COM-25, pp. 795-799, Aug. 1977.

[2] K. Yao,"Error probability of asynchronous SSMA communication systems", IEEE Trans., vol. COM-25, pp. 803-809, Aug. 1977.

[3] D. E. Borth, M. B, Pursley, D. V. Sarwate, W. E. Stark,"Bounds on error probability for DS-SSMA communications", MIDCON Conference, paper 15/1, Nov. 1979.

[4] M. B. Pursley, D. V. Sarwate, W. E. Stark,"On the average probability of error for DS-SSMA systems", Proceedings of the 1980 Conference on Information Sciences and Systems, Princeton University, Princeton, NJ, pp. 320-325, March 1980.

[5] K.-T. Wu, D. L. Neuhoff,"Average error probability for DS-SSMA communication systems", Proceedings of the 18th Annual Conference on Communication, Control and Computing, pp. 359-368, Oct. 1980.

[6] E. A. Geraniotis, M. B. Pursley,"Error probability for binary PSK SSMA communications", Proceedings of the 15th Annual Conference on Information Sciences and Systems, The Johns Hopkins University, Baltimore, MD, March 1981.

[7] H. Cramer, Mathematical Methods of Statistics, Princeton, NJ: Princeton University Press, 1966.

[8] R. S. Freedman,"On Gram-Charlier approximation", IEEE Trans., vol. COM-29, pp. 122-125, Feb. 1981.

[9] R. C. Hanlon, C. S. Gardner,"Error probability of DS-SSMA systems on nonselective fading channels", IEEE Trans., vol. COM-27, pp. 1696-1700, Nov. 1979.

[10] A. Papoulis, The Fourier Integral and its Application, McGraw-Hill, NY, pp. 227-239, 1962.

[11] M. B. Pursley, H. F. A. Roefs,"Numerical evaluation of correlation parameters for optimal phase of binary shift-register sequences", IEEE Trans., vol. COM-27, pp. 1597-1604, Oct. 1979.

[12] G. Szegö, Orthogonal Polynomials, American Mathematical Society, Providence, Rhode Island, 4th ed., 1978.

Number of Users

E_b/N_o (db)	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
5	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
6	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
7	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
8	3	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
9	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
10	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2
11	4	4	4	4	4	4	4	4	4	4	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2
12	4	4	5	5	5	4	4	4	4	4	4	4	4	4	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2	2
13	5	6	6	6	6	6	5	5	5	4	4	4	4	4	4	3	3	3	3	3	3	3	3	3	2	2	2	2	2	2
14	?	?	6	6	7	6	6	6	6	5	5	5	5	4	4	4	4	4	3	3	3	3	3	3	3	2	2	2	2	2
15	?	?	?	?	9	8	8	7	7	7	6	6	6	5	5	5	5	4	4	4	3	3	3	3	3	2	2	2	2	2

Table 1 List of the number of terms for the Gram-Charlier expansion method

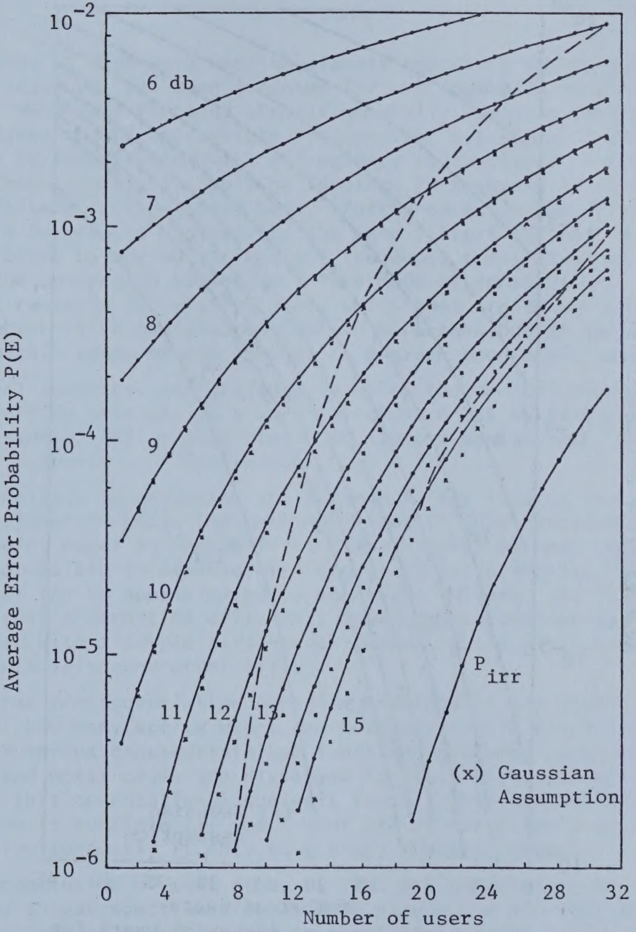


Fig. 1 P(E) versus number of users for 31 linear m-sequences

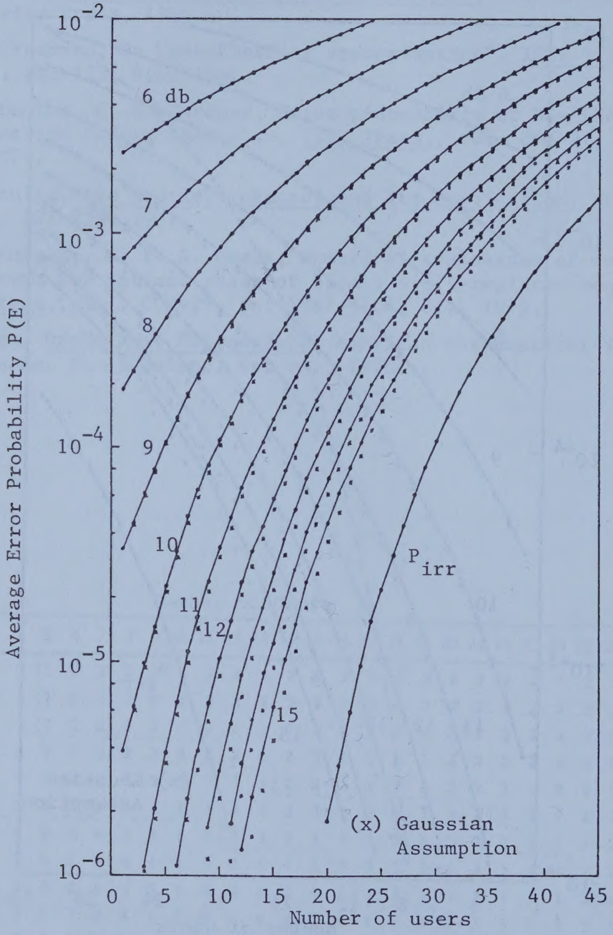


Fig. 2 P(E) versus number of users for Gold code

A SIMPLIFIED APPROACH TO THE DESIGN AND GENERATION OF SPREAD-SPECTRUM SIGNALS

G. R. COOPER, E. J. HEALD
School of Electrical Engineering
Purdue University
W. Lafayette, IN 47907

SUMMARY

A class of frequency-hopping signals employing short, periodically-repeated sequences has been proposed for use in mobile communication systems. Receivers for such signals generally employ a bank of bandpass filters tuned to the appropriate frequencies, but signal generation is likely to be accomplished with a frequency synthesizer. A method of generating these signals by applying impulses in sequence to an identical bank of filters is considered here. Furthermore, in the interest of economy, it is desired to use exactly the same filters for each user and make the user codes unique by appropriate frequency translation and rearrangement of the connection sequences. This mode of construction is suitable for short range (< 300 m), low power units that are to be mass produced, but for which it is anticipated that no more than one or two hundred units will be within range of one another in a given geographic area.

Signal sequences are designed by using the one-coincidence method and then converting this set to a zero-coincidence set by frequency translation. A typical design that yields 40 signals containing 7 hops each is shown for purposes of illustration.

An analysis to determine the optimum filter impulse response indicates that the signal-to-noise ratio is maximized if this function is symmetrical about a point equal to one-half the time interval between hops. Although this is impossible to achieve with causal infinite impulse response filters, it can be approximated by practical filters. It is shown, for example, that a detection efficiency (compared to the optimum) of .74 can be obtained with a simple first-order filter, while a value of .90 is achieved with a second-order filter.

The rms crosscorrelation function envelope of the signal set is a measure of how many active users can be supported in a multiple-access system. Numerous crosscorrelation function envelopes have been computed and best and worst cases are displayed for both first and second-order filters. This computational analysis reveals that the average crosscorrelation is sufficiently small that all of the signals in the set can be used simultaneously if they have equal received powers.

The conclusion of this study is that the proposed technique for generating spread-spectrum signals is a viable one and that the resulting signals perform at least as well as more conventional frequency-hopped signals.

ERROR PROBABILITY BOUNDS FOR UNBALANCED QAM OVER THE GAUSSIAN PHASE JITTERED, DISPERSIVE CHANNEL

T. L. LIM
Bell Laboratories
Holmdel, N.J. 07733

ABSTRACT

The purpose of this paper is to present an analysis concerned with a hybrid QAM system capable of handling simultaneously digital as well as analog data. Upper bounds are computed for the error probability in two and four-level digital data streams as a function of the noise power, linear distortion and phase jitter of the channel and of the relative power of the analog and digital signals. We show that the digital error probability is less than 10^{-5} if the noise and intersymbol interference in the digital channel are more than 24 dB below the signal level, the power of the quadrature analog signal differs by no more than 3 dB from that of the digital signal, the crosstalk from the analog into the digital channel is more than 40 dB down, and RMS phase jitter is at most 1 degree for the 4-level and 2 degrees for the 2-level digital signal.

ADAPTIVE EQUALIZATION AND PHASE TRACKING FOR SIMULTANEOUS ANALOG/DIGITAL DATA TRANSMISSION

T. L. LIM and M. S. MUELLER
Bell Laboratories
Holmdel, N.J. 07733

ABSTRACT

In some applications it is required to simultaneously transmit and receive digital and analog data. There are various ways to configure a modem for such a transmission scheme and here we describe how a standard QAM structure can be modified to operate in this mode. Our theoretical analysis shows that adaptive equalization and adaptive phase tracking can be achieved with the same quality as an all-digital modem. We provide extensive computer simulation results which confirm the validity of our theory.

1. INTRODUCTION

Recently, interest arose in finding ways to transmit and receive digital and analog data simultaneously over the 2-wire voiceband telephone channel. We investigate a system using Quadrature Amplitude Modulation (QAM) where digital and analog data modulate the two quadrature channels, also independently studied in [1].

The effects of various channel impairments, as well as an imbalance of signal powers between the digital and analog signals, on the error probability of the system has been studied in [2]. In this paper we analyze an adaptive equalizer for this type of hybrid modulation, using a cross-coupled transversal filter as described by Falconer in [3,4]. The difference here is that one of the two quadrature channels transmits analog data. Since references for the analog data are not available, especially when the receiver is in a decision directed mode, update algorithms reported in [3,4] are not applicable in this case. As an extension to [5,6], we analyze a technique using only the error signal from the digital data path of the equalizer.

Results from computer simulations are presented to verify our analytical results. The analog data actually acts as an interferer. As such, it would seem advantageous to reduce the analog signal power but this would degrade the analog signal-to-noise ratio (SNR).

2. MATHEMATICAL MODELS

The general QAM transmission scheme of Figure 1 is considered. Two data sequences $\{a_n\}$ and $\{b_n\}$, are applied to the inphase and quadrature inputs of the cross-coupled transmission filters. Their output signals are modulated by sine and cosine waves of carrier frequency ω_0 to form the passband signal s(t)

$$s(t) = \mathrm{Re}\left\{\sum_n D_n G(t-nT)e^{j\omega_0 t}\right\} \tag{1}$$

where T is the symbol interval, $D_n = a_n + jb_n$ and $G(t) = g_p(t) + jg_q(t)$. $\{D_n\}$ is the complex sequence of data symbols and G(t) is the complex impulse response of the transmission filter. We will report results for a system with $G(t)$ real and where $\{a_n\}$ and $\{b_n\}$ are digital and analog data sequences respectively. These sequences are assumed to have zero-mean, uncorrelated and $E\{a_n a_m\} = P_a \delta_{nm}$, $E\{b_n b_m\} = P_b \delta_{nm}$ where δ_{nm} is the Kronecker delta function.

After passing through a noisy, dispersive and phase jittered channel the signal at the input to the receiver can be expressed as

$$q(t) = \mathrm{Re}\left\{\exp[j(\omega_o t + \theta(t))] \sum_k D_k U(t-kT)\right\} + \eta(t)$$

where $U(t)$ is the overall complex impulse response of the transmitting filter and channel. The noise process $\eta(t)$ is independent of the data sequences while the phase shift $\theta(t)$ caused by the channel is assumed to vary much slower than the channel impulse response $h_c(t)$.

The general QAM receiver is depicted in Figure 2. The received signal $q(t)$ is bandpass filtered by the phase splitter pair, the outputs of which, at time kT, is

$$X_k = \exp[j(\omega_0 kT+\theta_k)] \sum_\ell D_\ell H_{k-\ell} + N_k \ , \tag{2}$$

where $\{H_k = H(\tau+kT)\}$ are the samples of the overall complex baseband equivalent impulse responses, and $\{N_k\}$ are the complex samples of the filtered noise process. The latter process is uncorrelated with the signal.

The complex signal sequence $\{X_k\}$ is passed through a cross-coupled passband equalizer with 2M+1 complex taps, the output of which at time nT is given by

$$Q_n = \sum_{k=-M}^{M} C_k^* X_{n-k} = \underline{C}^* \underline{X}_n \ , \tag{3}$$

when $\underline{C}^t = [C_{-M},\ldots,C_M]$ and $\underline{X}_n^t = [X_{n+M},\ldots,X_{n-M}]$. We use the '*' to denote conjugation for scalars and conjugate transposition for vectors and matrices. The symbol 't' denotes transposition.

The signal Q_n is demodulated to baseband by multiplication with $\exp(-j\hat{\theta}_n - j\omega_o nT)$ where $\hat{\theta}_n$ is the estimated phase offset (or jitter) at time nT. The resulting signal Y_n can then be written as

$$Y_n = Q_n \exp[-j\hat{\theta}_n - j\omega_o nT] = y_n + j\check{y}_n \ . \tag{4}$$

3. OPTIMIZATION OF EQUALIZER COEFFICIENTS

We define an optimum tap vector which minimizes the MSE in that branch where a reference signal is available or can be estimated easily. Assuming we have a reference for $\{a_n\}$ we define the MSE in that branch as the cost function to be minimized

$$\varepsilon_n^2 = E\{(y_n - a_n)^2\} \tag{5}$$

with
$$y_n = \mathrm{Re}\{\underline{C}^* \underline{X}_n \exp(-j\hat{\theta}_n - j\omega_o nT)\} \ ,$$

$$= C^t T(\Delta\theta_n) X_n \ , \tag{6}$$

where we partitioned the complex vectors $\underline{C}$ of the passband equalizer coefficients and $\underline{X}_n \exp(-j\theta_n - j\omega_o nT)$ of the ideally demodulated received signal to get real vectors C and X_n with twice the original dimension

$$C^t = (\mathrm{Re}\{\underline{C}^t\} \mid \mathrm{Im}\{\underline{C}^t\}) \tag{7}$$

$$X_n^t = \left[\mathrm{Re}\left\{\underline{X}_n^t \exp(-j\theta_n - j\omega_o nT)\right\} \middle| \mathrm{Im}\left\{\underline{X}_n^t \exp(-j\theta_n - j\omega_o nT)\right\}\right] \tag{8}$$

$$\Delta\theta_n = \theta_n - \hat{\theta}_n \ . \tag{9}$$

$T(\Delta\theta)$ is a transformation matrix expressing the effect of the demodulating phase error $\Delta\theta_n$ and can be shown to be

$$T(\alpha) = \left[\begin{array}{cc|cc} \cos\alpha & 0 & -\sin\alpha & 0 \\ & \ddots & & \ddots \\ 0 & \cos\alpha & 0 & -\sin\alpha \\ \hline \sin\alpha & 0 & \cos\alpha & 0 \\ & \ddots & & \ddots \\ 0 & \sin\alpha & 0 & \cos\alpha \end{array}\right] \tag{10}$$

Note that this matrix is orthogonal, i.e., $T(\alpha)\cdot T^t(\alpha) = T^t(\alpha)\cdot T(\alpha) = I$; $T(-\alpha) = T^t(\alpha)$; and $T(\alpha+\beta) = T(\alpha)\cdot T(\beta) = T(\beta)\cdot T(\alpha)$. In order to get the mean square error (5) as an explicit function of the coefficient vector C we introduce the autocorrelation matrix A of the demodulated received signal

$$A = E\left\{X_n X_n^t\right\} \tag{11}$$

and the cross correlation vector V between the demodulated received signal and the reference

$$V = E\left\{X_n a_n\right\} \ . \tag{12}$$

With (6), (11), and (12), we can express (5) as

$$\varepsilon_n^2 = C^t T(\Delta\theta_n) A T(-\Delta\theta_n) C - 2C^t T(\Delta\theta_n) V + E\left\{a_n^2\right\}. \tag{13}$$

Setting the partial derivatives with respect to C to zero yields the vector equation for the optimum tap vector C_{opt}

$$AT(-\Delta\theta_n)C_{opt} = V \tag{14}$$

The definition of A in (11) insures that it is positive definite. Consequently, the equation has a unique solution.

$$C_{opt} = T(\Delta\theta_n)A^{-1}V \ . \tag{15}$$

Inserting (15) into (13) yields for the minimum mean square error (MMSE)

$$\varepsilon^2_{opt} = E\left\{a^2_n\right\} - C^t_{opt}T(\Delta\theta_n)V = E\left\{a^2_n\right\} - VA^{-1}V. \tag{16}$$

When the power of the two data sequences is balanced, i.e., $P_a = P_b$, it can be shown that the resulting equation for the optimum tap vector coincides exactly with the equation resulting from minimizing the total MSE in both branches.

4. ANALYSIS OF THE GRADIENT ALGORITHM FOR JOINT EQUALIZATION AND PHASE TRACKING

In an adaptive receiver, the equalizer is assumed to know the reference data for startup and to operate in a decision-directed mode when random data is being sent. In either case, the tap weight vector is being updated continuously. Similarly, the phase offset, or jitter, would be continuously tracked in order to remain in synchronism. As in [3,4], we assume gradient algorithms updates,

$$C_{n+1} = C_n - \frac{\beta}{2}\nabla_C\varepsilon^2_n \tag{17a}$$

$$\hat{\theta}_{n+1} = \hat{\theta}_n - \frac{\alpha}{2}\frac{\partial}{\partial\hat{\theta}_n}\varepsilon^2_n \ , \tag{17b}$$

where $\nabla_C\varepsilon^2_n$ denotes the gradient of ε_n with respect to C.

It can be shown from (13) that

$$\nabla_C\varepsilon^2_n = 2T(\Delta\theta_n)[AT^t(\Delta\theta_n)C_n-V] \tag{18a}$$

and

$$\frac{\partial}{\partial\hat{\theta}_n}\varepsilon^2_n = 2C^t_n\frac{\partial T}{\partial\hat{\theta}_n}(\Delta\theta_n)[AT^t(\Delta\theta_n)C_n-V] \ . \tag{18b}$$

Using (11), (12), and (6), (18a) becomes

$$\nabla_{C_n} \varepsilon_n^2 = 2E\{T(\Delta\theta_n)X_n[y_n - a_n]\} \tag{19a}$$

$$\frac{\partial \varepsilon_n^2}{\partial \hat{\theta}_n} = 2E\left\{C_n^t \, \frac{\partial T(\Delta\theta_n)}{\partial \hat{\theta}_n} \, X_n[y_n - a_n]\right\} \tag{19b}$$

Finally, we use another property of the transformation matrix, $\frac{\partial T(\alpha+\beta)}{\partial \alpha} = T(\alpha + \beta\,\frac{\pi}{2})$ to obtain

$$\frac{\partial \varepsilon_n^2}{\partial \hat{\theta}_n} = 2E\{\check{y}_n[y_n - a_n]\} \quad . \tag{19c}$$

The equations (19a) and (19c) can be used to update the equalizer according to (17a,b).

Stochastic gradient update equations are obtained by omitting the expectations in (19) and making small corrections in the direction of the instantaneous values instead.

Inserting (19a) into (17a) yields the update equations

$$C_{n+1} = C_n - \beta'(y_n - a_n)T(\Delta\theta_n)X_n \ , \tag{20}$$

and

$$\check{\theta}_{n+1} = \check{\theta}_n - \frac{\alpha'}{a_n^2}\,\check{y}_n(y_n - a_n) \ . \tag{21}$$

It can be shown that for stability, the control loop constant has to satisfy

$$0 \le \beta' \le \frac{2}{E\left\{X_n^t X_n\right\}} \quad . \tag{22}$$

5. SIMULATION RESULTS

Here, we present simulation results for the hybrid modulation scheme. Pseudorandom digital data selected from the {±1, ±3} alphabet is used to modulate the inphase channel while a set of pseudorandom numbers with a Gaussian distribution $N(0,P_b)$ modulates the quadrature channel where P_b is the analog signal power. Additive Gaussian noise at -30 dB below the average signal level is introduced in the channel. The channels used in the simulation are (a) a good channel with a flat amplitude frequency response within most of the frequency band of interest and small delay distortion except near the lower band edge, and (b) a channel just violating the requirements for basic conditioning with both moderate amplitude and delay distortion. These characteristics are shown in Figure 3.

The first run described is for balanced power ($P_a=P_b=5$) where the receiver has a 64-tap T/2 complex equalizer. The Basic channel described above is used. Figure 4a is a sample simulation run displaying the SNR as a function of time where SNR is defined as the ratio of the digital signal power to the digital mean square error. The two curves are for two different timing phases in the receiver. As shown, the equalizer converges in about 2000 iterations for a step size of $\beta = 0.0005$. The receiver's digital and analog outputs after equalization and demodulation to baseband are displayed in Figure 4b as a scatter plot. The vertical and horizontal axes represent analog and digital values respectively. The decision thresholds for the digital data are the vertical lines with abscissae 0 and ±2. In this example, and all the others to be described, the input SNR is about 30 dB so the equalizer has done a reasonable job in removing intersymbol interference (ISI) caused by the Basic channel.

We next exhibit the results for the regular QAM in Figures 5a and 5b, with digital data on both branches and using exactly the same channel and receiver parameters as before. Convergence is faster than for the hybrid modulation since here we used a complex error function, leading to a tap weight vector $\underline{C}_n$ that fluctuates less than when using a one-sided error signal only.

We mentioned in Section 3a that when the powers in the digital and analog branches are balanced, the optimum equalizer and MSE are the same as that of a regular QAM receiver. Indeed, we see by comparing Figures 4a and 5a that the digital MSEs reach the same levels. It was also observed that the significant tap weights for both receiver timing phases differ by less than 5 percent.

Now we exhibit the effect on the mean square error of unbalancing the analog signal power from $P_b = 5$ to some other value. The simulated receiver has an AGC which scales the received signal to an average power of 'ten' and hence the equalizer tap adjustment step size is kept the same in this series of runs. The digital signal power is fixed at 'five' but the analog power is varied from 'one' to 'nine.' The output MSE normalized to the power of the corresponding signal is presented in Table 1 for the Good channel. As the analog power is increased, digital mean square error increases. It was also shown in [2] that the error probability upper bound is increased when a power imbalance exists in favor of the analog signal. As expected, the normalized analog mean square error is smaller with a larger analog signal power.

6. CONCLUSION

A model capable of transmitting and receiving analog and digital data simultaneously has been studied in detail. It is possible to perform adaptive equalization of the channel even when only one of the two quadrature channels is carrying digital data. Moreover, the minimum mean square error and tap weight vector are unchanged from that of the regular QAM as long as the analog and digital signal powers are equal.

REFERENCES

1. F. Akashi, Y. Sato, and M. Eguchi, "High Speed Digital and Analog Parallel Transmission Techniques Over a Single Telephone Channel," to be published.

2. T. L. Lim, "Error Probability Bounds for Unbalanced QAM Over the Gaussian, Phase Jittered, Dispersive Channel," 19th Allerton Conference, 1981.

3. D. D. Falconer, "Jointly Adaptive Equalization and Carrier Recovery in 2-Dimensional Digital Communication Systems," B.S.T.J., 55 (March 1976), pp. 317-34.

4. D. D. Falconer, "Analysis of a Gradient Algorithm for Simultaneous Passband Equalization and Carrier Phase Recovery," B.S.T.J., 55 (April 1976), pp. 409-28.

5. M. S. Mueller, "Complex Valued Analysis of Cross Coupled Equalizers in Synchronous Data Receivers," (in German) Swiss Federal Inst. of Technology, Ph.D. Thesis 5711, 1976.

6. G. Sandegren, "QAM and SSB Modulation for Data Transmission Over Telephone Lines Including Equalizer," Ericsson Technics, 33 (1977), pp. 247-98.

Table 1

Comparison of Analog and Digital

MSE for Different P_b

Analog Power P_b	Normalized MSE	
	Digital	Analog
1	0.0007	0.0040
3	0.0010	0.0017
5	0.0013	0.0013
7	0.0016	0.0011
9	0.0018	0.0009

Good Channel

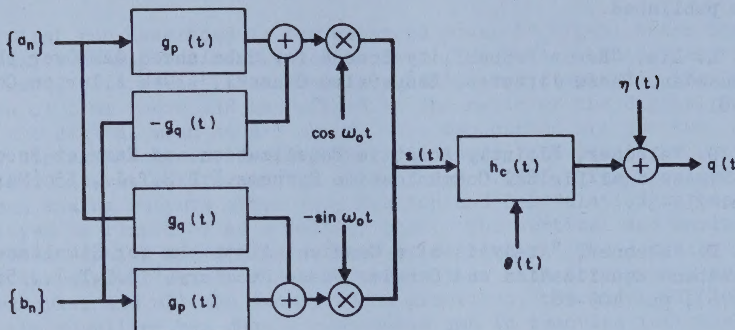

$\{a_n\}$
$g_p(t)$
$-g_q(t)$
$\cos \omega_o t$
$s(t)$
$\eta(t)$
$h_c(t)$
$q(t)$
$g_q(t)$
$-\sin \omega_o t$
$\theta(t)$
$\{b_n\}$
$g_p(t)$

FIGURE 1 MODEL OF TRANSMITTER AND CHANNEL

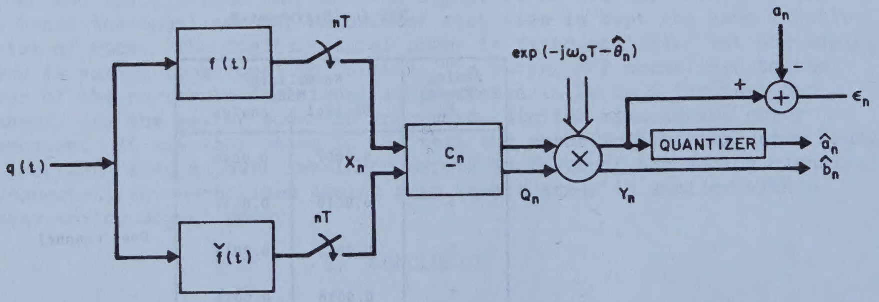
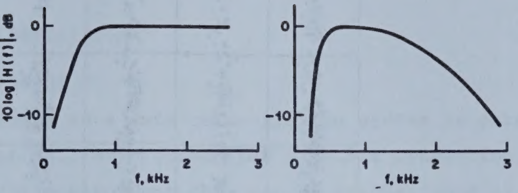

nT
$f(t)$
$\exp(-j\omega_o T - \hat{\theta}_n)$
a_n
$-$
$+$
ϵ_n
$q(t)$
X_n
$\underline{C}_n$
QUANTIZER
$\hat{a}_n$
$\hat{b}_n$
Q_n
Y_n
nT
$\check{f}(t)$

FIGURE 2 RECEIVER STRUCTURE



(a) GOOD CHANNEL (b) BASIC CHANNEL

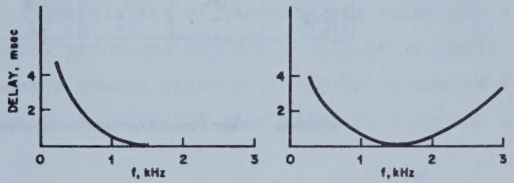


FIGURE 3 CHANNEL FREQUENCY RESPONSES

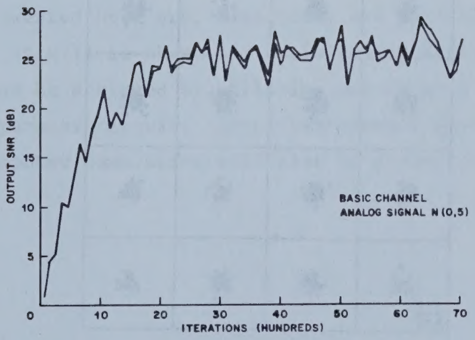


FIGURE 4a CONVERGENCE OF HYBRID EQUALIZER

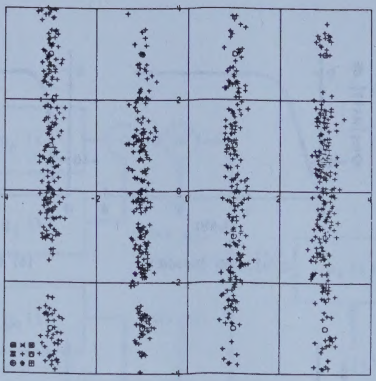


FIGURE 4b HYBRID RECEIVER OUTPUT CONSTELLATION

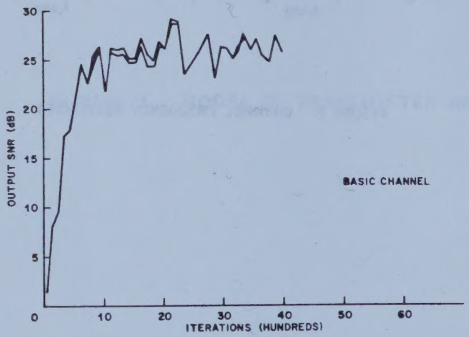


FIGURE 5a CONVERGENCE OF REGULAR QAM EQUALIZER

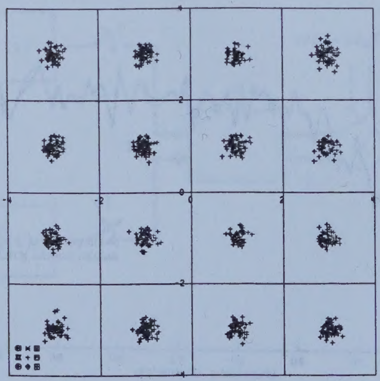


FIGURE 5b REGULAR QAM RECEIVER OUTPUT CONSTELLATION

CORRELATION TECHNIQUES APPLIED TO THE BIT SYNCHRONIZATION PROBLEM

C. R. GUARINO
HRB-Singer--System Definition
Science Park Road
State College, PA 16801

One fundamental concern in a data communication system is establishing bit synchronization, that is, the precise time of a bit transition. In a high signal-to-noise (S/N) environment this can be accomplished by sampling the incomming data line at a higher rate than the anticipated bit rate and searching for a transition. This high sampling rate enables detection of the transition time with a granularity of the sampling rate. In a low S/N environment, a simple search procedure may not result in suitable performance. In this paper a simple procedure will be presented for self-synchronizating systems which acquire bit sync directly from the incoming data sequence.

Often bit synchronization is only one function of a digital signal detection system. Figure 1 shows a possible integrated configuration of a bit synchronizer and detector. In a Frequency Shift Keying (FSK) system the bit synchronizer determines the start time of frequency measurement and inaccuracies in bit synchronization result in degraded frequency measurement performance. The interrelationships between FSK detector and performance is complicated and will not be considered in this paper except to note that certain detection schemes are less sensitive to synchronizer inaccuracies than others. It will be assumed that the bit synchronizer and detector can be isolated and evaluated independently.

Several methods for optimal bit synchronization have been presented in the literature. The major distinctions between previous approaches and the one to be presented here are, simplicity and digital approach to implementation. It will be shown in the following section that a digital implementation can be achieved by utilizing memory as a storage device, adders and a comparator circuit. Error performance which was determined via a digital computer simulation will also be presented.

A TYPE II HYBRID ARQ SCHEME FOR DATA TRANSMISSION IN MOBILE RADIO SYSTEMS

RICHARD A. COMROE
Motorola Inc.
Corporate Research
Schaumburg, Illinois 60196

DANIEL J. COSTELLO, JR.
Dept. of Electrical Engineering
Illinois Institute of Technology
Chicago, Illinois 60616

ABSTRACT: A new coding and retransmission protocol for data packet communications is described for use on Land Mobile Radio systems. The technique is called Type II Hybrid ARQ with Selective Retransmission and appears to offer significant sensitivity and throughput advantages over other known techniques.

There have been several approaches to delivering data packets between users on a common Land Mobile Radio (LMR) channel. Variations on a Stop-and-Wait ARQ strategy are most commonly used. In this approach the sender waits an appropriate time after the transmission of a packet for the receiver to transmit a positive acknowledgment. After waiting or receipt of a negative acknowledgment, the sender repeats the transmission until a positive acknowledgment is received. Because of the high error rate on a LMR channel this technique can be very inefficient requiring many retransmissions to deliver a message.

Another approach is to encode the packets in a Forward Error Correction (FEC) code. Although this significantly lengthens the message, it does increase the success probability of each transmission and thus reduces the number of retransmissions. This technique is called Type I Hybrid ARQ.

In another approached called Selective Retransmission the packet is first subdivided into smaller blocks. Error Detection (ED) bits are added to each block before the packet is transmitted. At the receiver each block can be checked, and only those blocks found to contain errors need be retransmitted. The acknowledgment in this approach carries a vector to identify to the sender which blocks need to be retransmitted. This technique reduces the number and the size of the retransmissions without increasing the initial packet length.

Selective retransmission can also be combined with Type I Hybrid ARQ where FEC coding is further applied to each block, which already contains ED bits, to increase the success probability of the block. This reduces further the number of retransmissions although it does increase the packet length.

A type II Hybrid ARQ scheme with Selective Retransmission obtains the advantages of the last approach without increasing the message length. This is accomplished by not sending the FEC bits in the first transmission. Then for those blocks which were not received error-free, only the FEC bits are retransmitted. For blocks where neither the first or second transmission was error-free, FEC can be attempted. Still failing to obtain correct decoding, further retransmissions alternate between data blocks and parity blocks.

In summary, Type II Hybrid ARQ with Selective Retransmission appears to achieve the advantages of FEC without the penalty of increased message length. On good channels, it acts like a high rate ED code. On poor channels, it utilizes FEC for only the parts of the message which require it.

This work was supported by the Motorola Corporate Research & Development Center, and by the National Science Foundation under grant ECS 8103746 at the Illinois Institute of Technology.

Optimal Coding for Information Transmission through a Poisson Type Channel

Aurel A. Lazar and Stuart C. Schwartz

Columbia and Princeton University

Abstract

The problem of optimal coding and decoding of an information bearing signal transmitted through a Poisson type channel is considered. As in the case of transmission of a Gaussian random process through a white Gaussian channel, it is shown that with respect to the m.m.s.e. criterion the optimal coding of an "n bit process" is linear; feedback, however is not required.

I. Introduction

The problem of optimal coding and decoding of an information bearing signal transmitted through a Poisson type channel is considered. Although of central importance for investigations related to optical communications, the problem has received little attention. By way of contrast, the similar problem of coding and decoding of a signal transmitted through the white Gaussian channel is fairly well understood by now. Specifically, there is an optimal solution for the transmission of Gaussian random processes through the white Gaussian channel. In [6] it is shown that the corresponding optimal coding is linear, requires feedback and is such that only the innovations process has to be transmitted, *i.e.*, white noise. These results hold for the minimum mean-square error or the maximum mutual information criterion. This problem was first studied for the case of transmitting a Gaussian random variable by Schalkwijk and Kailath [7]. Their result was extended by both Shiryayev [8] and Ihara [2] who also proved optimality with respect to the mean square error and maximum mutual information criteria, while a more general result is given in Liptser [5]. A good historical account of this problem can be found in Liptser and Shiryayev [6].

In studying the optimal coding and decoding of a stochastic process for transmission through the Poisson type channel, the first question which occurred was the following: what class of random processes play the same role for the Poisson type channel as does the class of Gaussian processes for the white Gaussian channel. By analogy with the coding of a Gaussian process through the white Gaussian channel, we "intuitively inferred" that processes taking values in a state space which contains only two elements (*e.g.* the random telegraph wave) play a fundamental role for the Poisson type channel. Foschini, Gitlin and Salz [1] also considered a binary signal model in their direct-detection optical communication receiver.

The problems investigated are generally formulated within the framework of an abstract probability space $(\bar{\Omega},\bar{F},\bar{P})$ $(\bar{\Omega}=\Omega\times\tilde{\Omega},\bar{F}=F\times\tilde{F})$ [3], [4].

Let $X=(X_t,F_t\times\tilde{F}_0)$, $0\leq t\leq T$, be an "information bearing signal" to be transmitted over a Poisson type channel. The observation process $N=(N_t,F_0\times\tilde{F}_t)$, $0\leq t\leq T$, at the channel output is assumed to be a counting point process satisfying the stochastic differential equation

$$dN_t=\lambda_t dt + dM_t\ , \qquad N_0=0, \tag{1}$$

for all t, $0 \le t \le T$, where λ_t is $F_t \times \tilde{F}_t$ measurable and $M=(M_t, F_t \times \tilde{F}_t)$, $0 \le t \le T$, is a martingale on the probability space $(\bar{\Omega}, \bar{F}, \bar{P})$.

The stochastic process $\lambda=(\lambda_t, F_t \times \tilde{F}_t)$, $0 \le t \le T$, called the sensor or random intensity rate (RIR) of the counting point process $\bar{N}=(N_t, F_t \times \tilde{F}_t)$, $0 \le t \le T$, determines the available knowledge about the process $X=(X_t, F_t \times \tilde{F}_0)$, $0 \le t \le T$. The above mathematical framework incorporates a variety of problems arising in practice, most notably those encountered in optical communications.

The aim of our investigation is to represent the RIR λ as a functional of the "information bearing process" X and the observations N, *i.e.*,

$$\lambda_t=\lambda(X_0^t, N_0^t), \tag{2}$$

for all t, $0 \le t \le T$, such that we attain or come close to some optimal properties. From (2) we adopt the phrase λ is the coding (of the process X).

The optimal properties in this paper concern the recovery of "information" contained in the "information carrying process" X that "modulates" the observations N. The coding λ is determined from among the class of admissible codes satisfying the peak constraint:

$$0 \le \lambda_t(X,N) \le c\ \ , \tag{3}$$

for all t, $0 \le t \le T$, where c, $c \in R_+$, is a constant.

Letting X be a "n bit process", it is shown that the optimal coding is linear as in the case of transmission of a Gaussian random process through a white Gaussian channel; feedback, however, is not required. More precisely the m.m.s.e. optimality criterion mentioned above is proven to be satisfied with the coding λ given by:

$$\lambda_t = c \cdot X_t, \tag{4}$$

for all t, $0 \le t \le T$.

II. Problem Setting

Definition 1. The nonanticipative functional $\lambda=(\lambda_t(X,N), F_t \times \tilde{F}_t)$, $0 \le t \le T$, will hereafter in this chapter denote the *coding*.

Definition 2. The class of codes satisfying the peak constraint [29], [13]:

$$0 \le \lambda_t(X,N) \le c\ \ , \tag{5}$$

for all t, $0 \le t \le T$, is called *admissible*.

Lemma 1. *Let the transmitted signal* $X=(X_t, F_t \times \tilde{F}_0)$, $0 \le t \le T$, *be a binomial random variable taking the values* a_0 *and* a_1 *with probability* $1-p$ *and* p, *respectively. Then there exist two positive nonanticipative functionals* $A_0=A_0(t,N)$, $0 \le t \le T$, *and* $A_1=A_1(t,N)$, $0 \le t \le T$, *such that* *the coding λ is completely described by*

$$\lambda_t(X,N) = A_0(t,N) + A_1(t,N)\frac{X_t-a_0}{a_1-a_0}, \tag{6}$$

for all t, $0 \leq t \leq T$.

Proof. Since

$$\lambda_t(X,N) = \begin{cases} \lambda_t(a_0,N) & \text{if } X_t = a_0 \\ \lambda_t(a_1,N) & \text{if } X_t = a_1, \end{cases}$$

for any t, $0 \leq t \leq T$, it is easy to see that the coding λ can be written as

$$\lambda_t(X,N) = \lambda_t(a_0,N) + [\lambda_t(a_1,N) - \lambda_t(a_0,N)]\frac{X_t-a_0}{a_1-a_0},$$

for all t, $0 \leq t \leq T$, and therefore there exist two positive nonanticipative functionals $A_0=A_0(t,N)$, $0 \leq t \leq T$, and $A_1=A_1(t,N)$, $0 \leq t \leq T$, such that (6) is satisfied. □

Note that $(X_t-a_0)/(a_1-a_0)$ is a binomial random variable taking the values 0 and 1 with probability $1-p$ and p respectively. Consequently, without any loss of generality we can assume that the signal $X=(X_t,F_t \times \tilde{F}_0)$, $0 \leq t \leq T$, to be transmitted is a random variable taking the values 0 and 1 with probability $1-p$ and p, respectively, and there exist two positive nonanticipative functionals $A_0=A_0(t,N)$, $0 \leq t \leq T$, and $A_1=A_1(t,N)$, $0 \leq t \leq T$, such that

$$\lambda_t(X,N) = A_0(t,N) + A_1(t,N)X_t, \tag{7}$$

for all t, $0 \leq t \leq T$.

Following the terminology of Liptser and Shiryayev [6] we state

Definition 3. The elements of the tuple (A_0,A_1) appearing in (7) are called the *coding functionals.*

In the following section we limit our study to the case where the information bearing signal process $X=(X_t,F_t \times \tilde{F}_0)$, $0 \leq t \leq T$, is an "n bit signal" ($n \in N$).

Defintion 4. X is said to be an *n bit process* if it is composed of n independent random variables on the intervals (0,T/n], (T/n,2T/n], ... , ((n-1)T/n, T] taking the values 0 and 1 with probability $1-p$ and p, respectively.

III. The Estimation Problem

At each instant of time t the output signal $\bar{X}_t(N)$ can be reconstructed from the received signal (N_s), $0 \leq s \leq t$, at the output of the processor.

The nonanticipative functional $\bar{X}=(\bar{X}_t(N),F_0\times\tilde{F}_t)$, $0\le t\le T$, specifying the decoding must be chosen to reproduce the signal X in the optimal manner.

Set

$$\Delta(t)=\inf \bar{E}[X_t-\bar{X}_t(N)]^2, \tag{8}$$

for all t, $0\le t\le T$, where inf is taken over all admissible codings $\lambda=(\lambda_s(X,N),F_s\times\tilde{F}_s)$, $0\le s\le t$, and decodings $\bar{X}=(\bar{X}_t(N),F_0\times\tilde{F}_t)$, for all t, $0\le t\le T$.

The problem is to find the optimal coding, decoding (if such exists), and the minimal reproduction error $\Delta(t)$ as a function of time. Since with a specified coding

$$\bar{E}[X_t-\bar{X}_t(N)]^2\ge \bar{E}[X_t-\bar{E}\{X_t\,|\,F_0\times\tilde{F}_t\}]^2,$$

it is seen that $\Delta(t)=\inf\bar{E}[X_t-\hat{X}_t]^2$, and that the optimal decoding (of signals $X=(X_t,F_t\times\tilde{F}_0)$, $0\le t\le T$) is the conditional mean estimate (CME) $\hat{X}_t=\bar{E}\{X_t\,|\,F_0\times\tilde{F}_t\}$. Thus the primary problem is reduced to the problem of finding only the optimal coding.

Remark. The minimization procedure required by (8) is equivalent to determining in Fig.1 the processor and the data recovery schemes from among all the schemes which can be built (implemented).

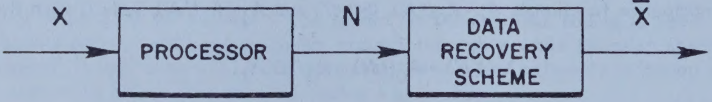


Fig. 1 Block diagram representation of the information processing scheme.

Definition 5. The coding $\lambda=(\lambda_t(X,N),F_t\times\tilde{F}_t)$, $0\le t\le T$, is said to be *optimum in the minimum mean square-error (m.m.s.e.) sense* if over the class of admissible codes the infimum

$$\inf_{0\le\lambda\le c}\bar{E}[X_t-\hat{X}_t(N)]^2, \tag{9}$$

is achieved for all t, $0\le t\le T$.

Theorem 1. *In the class of admissible codes the optimum coding in the m.m.s.e. sense exists and is given by*

$$\lambda_t=c\cdot X_t, \tag{10}$$

for all t, $0\le t\le T$.

The optimal decoding $\hat{X}_t$ *and the transmitted signal satisfy the equations*

$$\hat{X}_t = \begin{cases} \dfrac{1}{1+\frac{1-p}{p}e^{c(t-kT/n)}} & \text{if } N_t = 0 \\ 1 & \text{if } N_t \geq 1 \end{cases} \tag{11}$$

and

$$dN_t = c \cdot X_t \, dt + dM_t, \qquad N_0 = 0, \tag{12}$$

for all t and k, $kT/n < t \leq (k+1)T/n$ *and* $k=0,1,\cdots,n-1$.
The reproduction error is

$$\Delta(t) = \frac{1}{\frac{1}{1-p} + \frac{1}{p}e^{c(t-kT/n)}}, \tag{13}$$

for all t and k, $kT/n < t \leq (k+1)T/n$ *and* $k=0,1,\cdots,n-1$.

Proof. This assertion is proven by means of the principle of mathematical induction. Let us first consider that the information process $X=(X_t, F_t \times \tilde{F}_0)$, $0 \leq t \leq T$ is a "1 bit signal", *i.e.,* a binomial random variable taking the values 0 and 1 with probability $1-p$ and p, respectively. Using the abstract Bayes formula [6], the CME $\hat{X}_t$ becomes:

$$\hat{X}_t = E\left\{X_t \frac{q_t(\omega,\tilde{\omega})}{Eq_t(\omega,\tilde{\omega})}\right\},$$

where the sample function density of the counting point process $\bar{N}$ is specified by

$$q_t(\omega,\tilde{\omega}) = \exp\left[\int_0^t \ln(A_0(s,N) + A_1(s,N)X_s)dN_s - \int_0^t (A_0(s,N) + A_1(s,N)X_s)ds\right].$$

Thus

$$\begin{aligned}\hat{X}_t = {} & \{p\exp[\int_0^t \ln(A_0(s,N) + A_1(s,N))dN_s - \int_0^t (A_0(s,N) + A_1(s,N))ds]\} \\ & \cdot \{p\exp[\int_0^t \ln(A_0(s,N) + A_1(s,N))dN_s - \int_0^t (A_0(s,N) + A_1(s,N))ds] \\ & \qquad + (1-p)\exp[\int_0^t \ln A_0(s,N)dN_s - \int_0^t A_0(s,N)ds]\}^{-1},\end{aligned}$$

and therefore

$$\hat{X}_t = \frac{1}{1 + \frac{1-p}{p}\exp[\int_0^t \ln\frac{A_0(s,N)}{A_0(s,N) + A_1(s,N)}dN_s + \int_0 A_1(s,N)ds]}.$$

Note at this point that the likelihood ratio $\tilde{\Lambda}_t$ is given by

$$\tilde{\Lambda}_t = \exp\Big[\int_0^t \ln \frac{A_0(s,N) + A_1(s,N)}{A_0(s,N)} dN_s - \int_0^t A_1(s,N)ds\Big],$$

and thus

$$\tilde{E}[X_t - \hat{X}_t]^2 = p - \tilde{E}\hat{X}_t^2 = \tilde{E}[\hat{X}_t - \hat{X}_t^2]$$

$$= \int_{\tilde{\Omega}} \frac{1}{\frac{1}{1-p} + \frac{1}{p}\tilde{\Lambda}_t^{-1}} Q(0,d\tilde{\omega}).$$

Using inequality (5) it is easy to see that

$$\tilde{\Lambda}_t^{-1} = \exp\Big[-\int_0^t \ln \frac{A_0(s,N) + A_1(s,N)}{A_0(s,N)} dN_s + \int_0^t A_1(s,N)ds\Big] \le \exp\Big[\int_0^t A_1(s,N)ds\Big] \le e^{ct}.$$

Hence

$$\tilde{E}[X_t - \hat{X}_t]^2 \ge \frac{1}{\frac{1}{1-p} + \frac{1}{p}e^{ct}}, \tag{14}$$

and this lower bound can be achieved by setting

$$A_0(s,N) = 0 \quad \text{and} \quad A_1(s,N) = c, \tag{15}$$

for all s, $0 \le s \le T$, and therefore

$$\lambda_s(X,N) = c \cdot X_s, \tag{16}$$

for all s, $0 \le s \le t$. It is now easy to see that $\hat{X}_t$ is given by

$$\hat{X}_t = \frac{1}{1 + \frac{1-p}{p}e^{ct}}, \tag{17}$$

for all t, $0 \le t \le T$.

Remark. The result obtained can be interpreted as follows. In transmitting a random variable which takes only the values 0 and 1 under the peak constraint (5), one should set the energy on the signal in such a way that the two values it takes (0 and 1) are as far apart as possible. In estimating the signal X_t for any t, $0 \le t \le T$, we will be uncertain (commit an error) as long as the observation process does not have any arrivals ($N_t = 0$). In this situation feedback is not needed simply because there is nothing to feed back ($N_t = 0$). If, on the other hand, at least one arrival takes place we know with certainty that the value 1 of the signal was sent and therefore feedback is again not needed.

Let us extend the above result, in a natural way, by considering the process X as an "n bit signal".

Using an argument similar to the one in Lemma 1 it is easy to see that the most general description of the coding λ is given by

$$\lambda_t(X,N) = \sum_{k=0}^{n-1}[A_k(t,N) + A_{k+1}(t,N)X_t]\chi_{(kT/n,(k+1)T/n]}(t), \tag{18}$$

for all t, $0 \leq t \leq T$, where χ_B indicates the characteristic function of the set B. The elements of the n-tuple $(A_0, A_1, \cdots, A_{n-1})$ will be called the coding functionals (see also Definition 3).

Using the abstract Bayes formula we arrive, for all t and k, $kT/n < t \leq (k+1)T/n$ and $k = 0, 1, \cdots, n-1$, at

$$\hat{X}_t = E\left[X_t \frac{\exp\left(\int_{kT/n}^{t} \ln\lambda_s dN_s - \int_{kT/n}^{t} \lambda_s ds\right)}{E\exp\left(\int_{kT/n}^{t} \ln\lambda_s dN_s - \int_{kT/n}^{t} \lambda_s ds\right)}\right],$$

since the events on the intervals $\{(kT/n,(k+1)T/n]\}$, $k=0,1,\cdots,n-1$ are independent. Hence we have reduced our problem to the "1 bit signal" case discussed above. With respect to the m.m.s.e. criterion the optimal coding is given by

$$\lambda_t(X,N) = c \cdot X_t, \tag{19}$$

for all t, $0 \leq t \leq T$. Therefore

$$\hat{X}_t = \begin{cases} \dfrac{1}{1 + \frac{1-p}{p} e^{c(t-kT/n)}} & \text{for } N_t = 0 \\ 1 & \text{for } N_t = 1. \end{cases} \tag{20}$$

and

$$\bar{E}[X_t - \hat{X}_t]^2 = \frac{1}{1 + \frac{1-p}{p} e^{c(t-kT/2)}}, \tag{21}$$

for all t and k, $kT/n < t \leq (k+1)T/n$ and $k=0,1,\cdots,n-1$. □

Conclusion

In this paper the optimal coding for transmission of a "n bit signal" through a Poisson type channel has been investigated. With respect to the m.m.s.e. criterion the optimal coding of the "n bit signal" was proven to be linear. Furthermore it has been shown that the optimal coding does not require feedback.

Acknowledgement

The research reported here was supported in part by the National Science Foundation under grant #ECS 8106476, and in part by the Office of Naval Research under contract N00014-80-C-0530.

References

[1] Foschini, G.J., Gitlin, R.D. and Salz, J., "Optimum Direct Detection for Digital Fiber Optical Communication Systems", *The Bell System Technical Journal,* Vol. **No. 8, Oct. 1975.**

[2] Ihara, S., "Optimal Coding in White Gaussian Channel with Feedback", in *Proceedings of the Second Japan-USSR Symposium on Probability Theory,* Lecture Notes in Mathematics, No. **Springer Verlag, Berlin, 1973.**

[3] Lazar A.A., *Optimal Information Processing Using Counting Point Process Observations,* Ph.D. Dissertation, Dept. of Electrical Eng. and Computer Science, Princeton University, Princeton, NJ 08544, Oct. 1980.

[4] Lazar, A.A. and Schwartz, S.C., "Optimum Coding and Decoding of a Random Telegraph Wave for Transmission through a Poisson Type Channel", in *Proceedings of the International Symposium on Information Theory,* Santa Monica, CA, Feb. 9-12, 1981.

[5] Liptser, R.S., "Optimal Coding and Decoding in the Transmission of a Gaussian Markov Signal with Noiseless Feedback", *Problems of Information Transmission,* Vol. No. **4, pp. 3-15, Oct.-Dec. 1974.**

[6] Liptser, R.S. and Shiryayev, A.N., *Statistics of Random Processes II: Applications,* Springer Verlag, New York, 1978.

[7] Schalkwijk, J.P.M. and Kailath, T., "A Coding Scheme for Additive Noise Channels with Feedback - Part I: No Bandwidth Constraint", *IEEE Trans. Inf. Theory,* Vol. No. **2, April 1966.**

[8] Shiryayev, A.N., "Statistics of Diffusion Type Processes", in *Proceedings of the Second Japan-USSR Symposium on Probability Theory,* Lecture Notes in Mathematics, No. Springer Verlag, New York, 1973.

STOCHASTIC CONTROL OF TWO COMPETING QUEUES

JOHN S. BARAS AND STAVROS A. BELBAS
Electrical Engineering Department
University of Maryland
College Park, Maryland 20742

ABSTRACT

We consider optimal server time allocation to two parallel queues. The server has available complete past observations of the queue sizes for his decisions. The infinite time discounted version of the problem is analyzed here. It is shown that the optimal strategy is stationary. The optimal value function is shown to be the unique solution of the Bellman equation. Finally, analysis of degenerate Bellman equations, of the type appearing in this problem is presented. Numerical methods of solution can be derived from the results presented here.

1. INTRODUCTION

We consider the problem of selecting which of two parallel queues to serve with a single server. The system is depicted in Figure 1. Customers arrive into stations 1 and 2 according to two independent Poisson streams with constant rates λ_1, λ_2 respectively. The two queues compete for the services of an exponential server with constant service rate μ. Let $x_{i,t}$ be the number of customers in queue i at time t, the customer in service included. The control to be selected is clearly of a switching type. When $u_t=1$ and the server completes a service the next customer to be served comes from queue 1, while if $u_t=0$ the next customer comes from queue 2. The queue that is being served forms an M/M/1 system with the server. This is a simple sequencing problem, where the sequencing variable is u_t. Let $x_t=(x_{1,t},x_{2,t})$ denote the state at time t. u_t is to be selected knowning x_t.

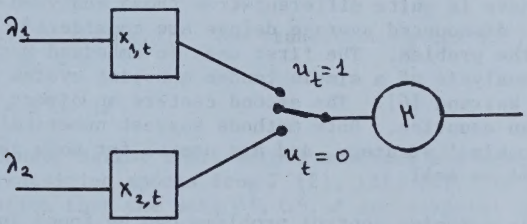


Fig. 1 The server time allocation problem

The server allocation time is to be selected to minimize delays, weighted according to c_1, c_2, two positive constants. Thus the cost per unit time with queues $x_{1,t}$, $x_{2,t}$ is $c_1x_{1,t}+c_2x_{2,t}=c^Tx_t$. It is shown that the policies which minimize the infinite time discounted average cost are characterized by a switching curve $S:x_2=S(x_1)$. That is $u_t=1$ if $x_{2,t}<S(x_{1,t})$, while $u_t=0$ qhwn $x_{2,t}>S(x_{1,t})$. Thus the optimal strategy is stationary.

The present paper is an outgrowth of earlier work [1] by one of the authors.

Research supported in part by the US Department of Energy under contract DE-AC01-79ET 29244.

There are two important differences between the work presented in [1] and the problems treated here. First the queues are observed here while only partial observations were assumed in [1]. Second finite time problems were treated in [1] while infinite time discounted problems are considered here.

The problem described can be thought of as a dynamic priority (non-preemptive) problem in a single queue with two classes of customers [2] [3]. The optimal static priority assignment (i.e. open loop control) which minimizes average cost per unit time, under steady state conditions was obtained by Cox and Smith [2]. The result, in the terminology used here, is that queue 1 has priority over queue 2 iff $c_1 > c_2$ (the opposite holds when $c_2 < c_1$). Actually their result is easily extended to include general service time distributions, as well as different distributions for different classes and to general m queues (classes) problems. Dynamic priorities minimizing average cost per unit time, under steady state conditions were derived by Rykov and Lembert [3]. The analysis presented in [3] is rather formal, however the results appear to be correct. The result, in the terminology used here, is that the optimal dynamic (feedback) priority coincides with the optimal static priority of Cox and Smith [2]. The result is valid under general service time distributions. In both cases the results were obtained by classical queueing methods and are in agreement with intuition, given the steady state assumptions and the average cost per unit time criterion used.

The problem considered here is quite different from those analyzed in [2] [3], since infinite time discounted average delays are considered. We use two methods to analyze the problem. The first uses an embedded Markov chain and is inspired by the analysis of a simple tandem queueing system studied by Rosberg, Varaiya and Walrand [6]. The second centers on direct analysis of the associated Bellman equation. Both methods suggest numerical schemes for computation of the optimal strategy, and are useful for more general queueing systems problems as well.

Other examples of dynamic queueing control problems can be found in [4] [5].

2. A METHOD BASED ON THE ASSOCIATED EMBEDDED MARKOV CHAIN PROBLEM

We consider the discounted continuous time problem for the two competing queues system described in figure 1. Let $\alpha \geq 0$ be the discount rate and

$$V_t^\alpha(x) := \min_{\gamma \in \Gamma} E\{\int_0^t c^T x_\sigma \, e^{-\alpha\sigma} d\sigma\} \tag{2.1}$$

be the minimum total expected discounted delay, when the time horizon is $t \geq 0$ and the starting state at time 0 is $x_0 = x$. In (2.1) Γ is the set of all admissible policies

$$\Gamma := \{\gamma = \{u_t(.), t \geq 0\} \text{ such that } u_t : Z\times Z \to \{1,0\}\} \tag{2.2}$$

where $Z = \{0,1,2,3...\}$.

The main objective is to characterize the optimal cost for the infinite time problem V_∞^α and the corresponding optimal strategy γ_∞^α. Considering the sizes of both queues x_t, as the state of the system, we have a Markov chain with countable state space $Z\times Z$. The transitions of x_t are easy to describe. They correspond to arrivals at queue 1 or queue 2, and to service completions. The transitions induced by arrivals occur at arrival times in queues 1 and 2 respectively denoted by

$$\begin{aligned}\mathcal{A}^1 &= \{t_0^1, t_1^1, t_2^1, \ldots\} \\ \mathcal{A}^2 &= \{t_0^2 m\ t_1^2, t_2^2, \ldots\}\end{aligned} \tag{2.3}$$

The control strategy does not affect arrivals. Therefore if we let A_1, A_2 be functions describing the new state after a transition induced by arrivals we see that for u = 1 or 0, the transition probabilities are

$$\left.\begin{aligned}A_1(x_1,x_2) &= (x_1+1,x_2) \quad \text{with Prob. } \lambda_1 dt + o(dt) \\ A_2(x_1,x_2) &= (x_1,x_2+1) \quad \text{with Prob. } \lambda_2 dt + o(dt\end{aligned}\right\} \tag{2.4}$$

On the other hand transitions induced by service completions occur at service completion times

$$\mathcal{S} = \{t_0^s, t_1^s, t_2^s, \ldots\} \tag{2.5}$$

and depend on the value of the control. Thus when u=1, D_1 is possible where

$$D_1(x_1,x_2) = ((x_1-1)^+,x_2) \text{ with Prob. } \mu dt + o(dt), \tag{2.6}$$

while when u = 0, D_2 is possible where

$$D_2(x_1,x_2) = (x_1,(x_2-1)^+) \text{ with Prob. } \mu dt + o(dt) \tag{2.7}$$

Here $(x)^+ = \max\{x,0\}$. The transition epochs [2], [3], [6], for the state process x_t are

$$\mathcal{T} = \mathcal{A}^1 \cup \mathcal{A}^2 \cup \mathcal{S} \tag{2.8}$$

The embedded Markov chain method analyzes the behavior of the state process x_t at transition epochs from $\mathcal{T}$ [2], [3], [6]. It is a consequence of our formulation that the sets $\mathcal{A}^1$, $\mathcal{A}^2$, $\mathcal{S}$ are disjoint (i.e. no simultaneous occurrence is possible).

To treat the infinite time discounted problem we need some results of Lippman [8], particularly since we do not have bounded costs for all x. The action space in the problem treated here is finite: {0,1}. The set of states accessible from a given state (x_1,x_2) in one transition is $\{(x_1+1,x_2), (x_1,x_2+1), ((x_1-1)^+,x_2), (x_1,(x_2-1)^+), (x_1,x_2)\}$, independent of the transition epoch. Furthermore the cost is linear in the state. As a consequence Assumptions 1 and 2 in Lippman [8, pp. 719] hold.

The strategies in Γ are Markovian [9]. Let us also consider the more general class of randomized nonanticipative strategies:

$$\tilde{\Gamma} := \{\gamma = (\pi_t(0), \pi_t(1)),\ t \geq 0\} \quad \text{such that } \pi_t \text{ is a function of the past history } h^t = (x_s, u_s, 0 \leq s \leq t). \tag{2.9}$$

Here $\pi_t(i)$, i = 0,1 is the probability of choosing u=0, or 1 at time t. A strategy in Γ is stationary whenever $u_t(.)$ is independent of t. We now have the following result:

Theorem 1:

The infinite time discounted problem (α>0), for the two competing queues, has an optimal strategy over $\tilde{\Gamma}$, which is Markovian and stationary. Furthermore the optimal value function V_∞^α is the unique solution of the Bellman equation

$$V^{\alpha}_{\infty}(i_1,i_2) = \frac{c_1}{\mu+\alpha}\, i_1 + \frac{c_2}{\mu+\alpha} + \frac{\mu}{\mu+\alpha} \min_{v\varepsilon\{0,1\}} \{\sum_{j_1,j_2} P^1_{i_1,j_1}(v) P^2_{i_2,j_2}(v) V^{\alpha}_{\infty}(j_1,j_2)\} \tag{2.10}$$

$$i_1 i_2 \geq 0,$$

where $P^1_{i_1,j_1}(v)$, $P^2_{i_2,j_2}(v)$ are given in (2.21) below.

Proof: Since Assumptions 1, 2 of Lippman [8] are satisfied and the action set is finite, the first result is an immediate consequence of Theorem 1 in Lippman [8, p. 719]. Since the optimal policy is stationary we can restrict consideration to stationary Markovian strategies which change values only at service completion times, $t^s_k \in \mathscr{S}$. We denote this restricted class by Γ_0. As a result we need find the equivalent discrete time stochastic control problem for the embedded Markov chain with transition epochs $\mathscr{S}$ and not $\mathscr{T}$. For ease of notation we denote x_{t_k} by x_k and u_{t_k} by u_k. It is plain that the intertransition intervals $t^s_{k+1} - t^s_k$ are independent, identically distributed random variables with an exponential distribution

$$\text{Prob}\ \{t^s_{k+1} - t^s_k > t\} = \exp\ (-t\mu). \tag{2.11}$$

Consider now a policy in Γ_0 and let us compute the corresponding cost over the random interval $[0,t^s_n]$ and initial state x:

$$E_x \int_0^{t^s_n} e^{-\alpha t} c^T x_t \, dt = E_x \sum_{k=0}^{n-1} \int_{t^s_k}^{t^s_{k+1}} e^{-\alpha t} c^T x_k \, dt$$

$$= E_x \sum_{k=0}^{n-1} \{c^T x_k\, E \int_{t^s_k}^{t^s_{k+1}} e^{-\alpha t} dt\} =$$

$$= \frac{1}{\alpha} E_x \sum_{k=0}^{n-1} \{c^T x_k E[e^{-\alpha t^s_k} - e^{-\alpha t^s_{k+1}}]\}$$

$$= \frac{1}{\alpha} E_x \sum_{k=0}^{n-1} c^T x_k (\beta^k - \beta^{k+1}) = \frac{1-\beta}{\alpha} E_x \sum_{k=0}^{n-1} \beta^k c^T x_k\ , \tag{2.12}$$

provided $\beta<1$, while when $\beta=1$ it equals $E_x \sum_{k=0}^{n-1} c^T x_k$. In the sequel we shall ignore the constant $\frac{1-\beta}{\alpha}$ in (2.12). Here

$$\beta = \frac{\mu}{\alpha+\mu}, \tag{2.13}$$

so $\beta=1$, corresponds to the discount rate $\alpha=0$. So the infinite time cost for the embedded Markov chain is

$$W^{\beta}_{\infty}(\gamma,x) = E_x\{\sum_{k=0}^{\infty} \beta^k c^T x_k\} \tag{2.14}$$

when policy $\gamma\varepsilon\Gamma_0$ is used. Obviously for $\alpha>0$,

$$V^{\alpha}_{\infty} = \frac{1-\beta}{\alpha} W^{\beta}_{\infty} \tag{2.15}$$

where W^{β}_{∞} is the optimal over $\gamma\epsilon\Gamma_0$ of (2.14). We next compute the transition probabilities for the embedded Markov chain x_k. Let $\xi_{1,k}$, $\xi_{2,k}$ be the total arrivals for queues 1, 2 during the interval $[t^s_k, t^s_{k+1}]$. Note that given $t^s_{k+1} - t^s_k = \delta$, $\xi_{1,k}, \xi_{2,k}$ are independent random variables, with Poisson distributions, with means $\lambda_1\delta$, $\lambda_2\delta$ respectively. Let $N_{1,t}$, $N_{2,t}$ be the counting processes for queues 1,2.

Then

$$\Pr\{N_{1,k+1}-N_{1,k}=\xi_{1,k}\} = \int_0^{\infty} \frac{e^{-\lambda_1\delta}(\lambda_1\delta)^{\xi_{1,k}}}{\xi_{1,k}!}\,\mu e^{-\mu\delta}\,d\delta$$

$$= \left(\frac{\lambda_1}{\lambda_1+\mu}\right)^{\xi_{1,k}} \frac{\mu}{\lambda_1+\mu} \tag{2.16}$$

Similarly

$$\Pr\{N_{2,k+1}-N_{2,k}=\xi_{2,k}\} = \left(\frac{\lambda_2}{\lambda_2+\mu}\right)^{\xi_{2,k}} \frac{\mu}{\lambda_2+k}\ . \tag{2.17}$$

Now if $u_k = 1$

$$\left.\begin{aligned} x_{1,k+1} &= x_{1,k} - 1 + \xi_{1,k}\ , && \text{if } x_{1,k}\neq 0 \\ x_{1,k+1} &= \xi_{1,k}\ , && \text{if } x_{1,k} = 0 \\ x_{2,k+1} &= x_{2,k} + \xi_{2,k} && . \end{aligned}\right\} \tag{2.18}$$

Similarly if $u_k = 0$

$$\left.\begin{aligned} x_{1,k+1} &= x_{1,k} + \xi_{1,k} && \\ x_{2,k+1} &= x_{2,k} - 1 + \xi_{2,k}\ , && \text{if } x_{2,k}\neq 0 \\ x_{2,k+1} &= \xi_{2,k}\ , && \text{if } x_{2,k} = 0 \end{aligned}\right\} \tag{2.19}$$

So the (reduced) embedded Markov chain has state space ZxZ and transition probabilities

$$P_{i_1,i_2,j_1,j_2}(u) := \Pr\{x_{k+1}=(j_1,j_2)\,|\,x_k=(i_1,i_2),u_k = v\}$$

$$= \Pr\{x_{1,k+1}=j_1\,|\,x_{1,k} = i_j,\ u_k = v\}.\Pr\{x_{2,k+1}=j_2\,|\,x_{2,k}=i_2,u_k=v\} \tag{2.20}$$

$$= P^1_{i_1,j_1}(v)\ P^2_{i_2,j_2}(v)$$

From (2.16) - (2.19) there result

$$P^1_{i_1,j_1}(1) = \begin{cases} (\lambda_1/\lambda_1+\mu)^{j_1}(\mu/\lambda_1+\mu)\ , & \text{if } i_1 = 0,\ j_1\geq 0 \\ (\lambda_1/\lambda_1+\mu)^{j_1+1-i_1}(\mu/\lambda_1+\mu)\ , & \text{if } i_1\geq 1,\ j_1\geq i_1-1 \\ 0\ , & \text{otherwise} \end{cases} \tag{2.21a}$$

$$P^1_{i_1,j_1}(0) = \begin{cases} (\lambda_1/\lambda_1+\mu)^{j_1-i_1}(\mu/\lambda_1+\mu)\ , & \text{if } j_1\geq i_1 \\ 0\ , & \text{otherwise} \end{cases} \tag{2.21b}$$

$$P^2_{i_2,j_2}(1) = \begin{cases} (\lambda_2/\lambda_2+\mu)^{j_2-i_2}(\mu/\lambda_2+\mu) & , \quad \text{if } j_2 \geq i_2 \\ 0 \quad , \quad \text{otherwise} & \end{cases} \tag{2.21c}$$

$$P^2_{i_2,j_2}(0) = \begin{cases} (\lambda_2/\lambda_2+\mu)^{j_2}(\mu/\lambda_2+\mu) & , \quad \text{if } i_2=0,\ j_2 \geq 0 \\ (\lambda_2/\lambda_2+\mu)^{j_2+1-i_2}(\mu/\lambda_2+\mu) & , \quad \text{if } i_2 \geq 1,\ j_2 \geq i_2-1 \\ 0 \quad , \quad \text{otherwise} \quad . & \end{cases} \tag{2.21d}$$

Then W^β_∞ is the unique solution of

$$W^\beta_\infty(i_1,i_2) = c_1 i_1 + c_2 i_2 + \beta \min_{v \varepsilon \{0,1\}} \left(\sum_{j_1,j_2} P_{i_1,i_2,j_1,j_2} W^\beta_\infty(j_1,j_2) \right) \tag{2.22}$$

Multiplying (2.22) by $\frac{1-\beta}{\alpha}$ we obtain (2.10), and this completes the proof of the theorem.

It is now clear how to obtain the switching curve in ZxZ, once V^α_∞ is known:

$$S = \{(i_1,i_2) \varepsilon ZxZ: \sum_{j_1,j_2} P^1_{i_1,j_1}(1) P^2_{i_1,j_2}(1) V^\alpha_\infty(j_1,j_2) = \sum_{j_1,j_2} P^1_{i_1,j_1}(0) P^2_{i_2,j_2}(0) V^\alpha_\infty(j_1,j_2)\} \tag{2.23}$$

From (2.21) easily follows that $(0,0)\varepsilon S$. The optimal stationary policy $\gamma^\alpha_\infty = \{f^\alpha, f^\alpha, \ldots\}$ is determined by the function $f^\alpha : ZxZ \to \{0,1\}$ as follows. If (i_1,i_2) is such that

$$\sum_{j_1,j_2} P^1_{i_1,j_1}(1) P^2_{i_2,j_2}(1) V^\alpha_\infty(j_1,j_2) < \sum_{j_1,j_2} P^1_{i_1,j_1}(0) P^2_{i_2,j_2}(0) V^\alpha_\infty(j_1,j_2) \tag{2.24a}$$

then

$$f^\alpha(i_1,i_2) = 1 \tag{2.24b}$$

and

$$f^\alpha(i_1,i_2) = 0 \tag{2.24c}$$

when the inequality in (2.24) is reversed. From (2.21), (2.24) easily follows that $f^\alpha(0,i_2) = 0$ and $f^\alpha(i_1,0) = 1$. These results agree with intuition.

The result of Theorem 1, suggests the following scheme for the numerical computation of V^α_∞ and f^α.

Let

$$V^\alpha_1(i_1,i_2) = \frac{c_1}{\mu+\alpha} i_1 + \frac{c_2}{\mu+\alpha} i_2 \quad , \quad i_1,i_2 \geq 0 \tag{2.25}$$

and define V^α_n recursively via

$$V_n^\alpha(i_1,i_2) = \frac{c_1}{\mu+\alpha}\, i_1 + \frac{c_2}{\mu+\alpha}\, i_2 + \frac{\mu}{\mu+\alpha} \min_{v\varepsilon\{0,1\}} \{\sum_{j_1,j_2} P^1_{i_1,j_1}(v) P^2_{i_2,j_2}(v)\, V^\alpha_{n-1}(j_1,j_2)\} \tag{2.26}$$

Let

$$f^\alpha_n : Z\text{x}Z \to \{0,1\}$$

be the function defined by the minimization in (2.26). Then we can show that

$$\left.\begin{aligned} &\lim_{n\to\infty} V^\alpha_n = V^\alpha_\infty \\ &\lim_{n\to\infty} f^\alpha_n = f^\alpha . \end{aligned}\right\} \tag{2.27}$$

3. DIRECT ANALYSIS OF THE BELLMAN EQUATION

First recall from [10] the following result on Bellman equations over an arbitrary Hilbert lattice V:

$$\max_{a\varepsilon\tilde{A}} (L^a u - f^a) = 0 \quad , \ u\varepsilon V,\ f^a \varepsilon V' \tag{3.1}$$

where L^a is coercive, and $\tilde{A}$ is the action set.

Let

$$K = \{v\varepsilon V : L^a v \leq f^a \quad , \quad \text{all } a\varepsilon\tilde{A}\} \ . \tag{3.2}$$

Next for arbitrary $a\varepsilon\tilde{A}$, let φ be the solution of the variational inequality

$$\langle L^a\varphi - f^a, v-\varphi\rangle \geq 0 \quad \text{for all } v\varepsilon K, \varphi\varepsilon K. \tag{3.3}$$

Then (see [10]) φ is the maximal element of the set of all subsolutions of (3.1) (which implies that φ is independent of the choice of a from $\tilde{A}$ in (3.2)). Furthermore, under appropriate technical assumptions [10], φ is also the strong solution of (3.1). In view of the result of Theorem 1 in section 2, it is only necessary to consider stationary problems.

For the case of interest here $\tilde{A} = \{1,0\}$, i.e. we have two operators L_1, L_0. Furthermore we have two state variables x_1, x_2 which are integer valued. Let us define

$$f(i_1,i_2) = \frac{c_1}{\mu+\alpha}\, i_1 + \frac{c_2}{\mu+\alpha}\, i_2 \ , \ i_1, i_2 \geq 0 . \tag{3.4}$$

Furthermore for a function $v(i_1,i_2)$ let

$$\left.\begin{aligned} &[L^0 v](i_1,i_2) = v(i_1,i_2) - \frac{\mu}{\mu+\alpha} \sum_{j_1,j_2}^{\infty} P^1_{i_1,j_1}(0) P^2_{i_2,j_2}(0) v(j_1,j_2) \\ &[L^1 v](i_1,i_2) = v(i_1,i_2) - \frac{\mu}{\mu+\alpha} \sum_{j_1,j_2}^{\infty} P^1_{i_1,j_1}(1) P^2_{i_2,j_2}(1) v(j_1,j_2). \end{aligned}\right. \tag{3.5}$$

Then (2.10) can be written

$$\max\,(L^0 v, L^1 v) = f \tag{3.6}$$

The main difficulty arising here is due to the fact that L_0, L_1 are not coercive, but degenerate. Thus, it is necessary to modify the method of [10] to apply to degenerate problems; this is done in [11]; here we present the method for the particular case of two operators.

It is easier to explain the main features of this method by introducing some functional spaces. Because our arguments do not depend on the maximum principle, they are applicable to both, the continuous and the discrete problems.

Let $\{1,2,\dots,n\} = A \cup B$, where n is the dimension of the spatial variable and $A \cap B \neq \phi$. Let $\Omega = \Omega_A \times \Omega_B \subseteq R^n$, where $\Omega_A \subseteq R^A$ and $\Omega_B \subseteq R^B$ are bounded open sets in the respective spaces. (The case of an unbounded set Ω can be handled similarly, by introducing weighted functional spaces).

Let

$$L_1 \equiv L_{1,1} + L_{1,2},\ L_0 \equiv L_{0,1} + L_{0,2}$$

where $L_{1,1}$ is a first-order hyperbolic (differential or finite difference) operator in the A-variables and $L_{1,2}$ is coercive in the B-variables, while the situation is reversed for L_0.

Now let

$$W_A = H_0^1(\Omega_A),\ W_B = H_0^1(\Omega_B)$$

$$\tilde{D}_A = \{v \varepsilon L^2(\Omega):\ v|\langle L_{1,1}, n\rangle|^{\frac{1}{2}}\ \varepsilon L^2(\partial\Omega_A),\ v = 0 \text{ on } \Gamma_A^-\}$$

where

$$\Gamma_A^- = \{x \varepsilon \partial\Omega_A:\ \langle L_{1,1}, n\rangle < 0\}.$$

$\tilde{D}_B$ is defined similarly.

(Notation: if $L = \sum_{i=1}^{m} b_i \frac{\partial}{\partial x_i} + b_0$, then $\langle L, n\rangle = \Sigma b_i n_i$; $n(x)$ is the outer normal at point $x \varepsilon \partial\Omega_A$).

Finite differences must be substituted instead of derivatives for the discrete problem. Further we denote by H_A, H_B the spaces $L^2(\Omega_A)$, $L^2(\Omega_B)$ (or $\ell^2(\Omega_A)$, $\ell^2(\Omega_B)$, for finite difference equations). We define the spaces

$$V_1 = \tilde{D}_A \otimes W_B,\quad V_0 = W_A \otimes \tilde{D}_B;\ U_1 = H_A \otimes W_B,\ U_0 = W_A \otimes H_B.$$

(Notation: $\otimes$ signifies tensor product.)

Let the convex set K be defined by

$$K = \{v \varepsilon V_1 \cap V_0:\ L_1 v \leq f_1,\ L_0 v \leq f_0\}.$$

We observe that K is sup-stable, i.e. u,wεK implies u$\cup$wεK.

Let $\bar{K}_1$ be the closure of K in U_1, and $K_1 = \bar{K}_1 \cap V_1$.

Let u be the solution of the variational inequality

$$\langle L_{1,1}v + L_{1,2}u - f_1,\ v - u\rangle$$
$$-\frac{1}{2}\int_{\Omega}[\nabla.L_{1,1}]\,(v-u)^2dx - \frac{1}{2}\int_{\Gamma_A^- \times\Omega_B}\langle L_{1,1}, n_A\rangle v^2 d\sigma \geq 0$$
$$\text{for all } v\epsilon K_1;\ u\epsilon \overline{K}_1 \tag{3.3}$$

Then we have the following:

Theorem 2:

There exists a solution u of (3.3) and is equal to the maximal element of $\overline{K}_1$.

Existence follows by a penalization argument as in [12]. The claim that u is the maximal element of $\overline{K}_1$ follows by methods similar to those of [10].

When $f^a \geq 0$, we have the following iterative scheme for computing the solution of (2).

Let u_0^a, a = 1,0 be subsolutions of (1), i.e. $u_0^a \epsilon V$, $L_a u_0^a \leq f^a$. Suppose that $u_0^a \epsilon \ell^\infty(\Omega)$. Let u_n^a be defined inductively as the solutions of the variational inequalities

$$\langle L_a u_n^a - f^a,\ v - u_n^a\rangle \geq 0 \text{ for all } v\epsilon \overline{K}_n^a;\ u_n^a \epsilon K_n^{a'} \tag{3.4}$$

where $K_n^a = \{v\ V_a : v \geq u_{n-1}^{a+1}\}$,

$\overline{K}_n^a = \{\text{closure of } K_n^a \text{ in } U_a\}$,

$K_n^{a'} = \overline{K}_n^a \cap V_a$

$a = 1,0.$

The variational inequalities (3.4) have strong solutions if we make the regularity assumption:

(A) $u_n^a \epsilon V_{a+1}$

Conditions under which the regularity assumption (A) is true are given in [13].

Then, we have the following:

Theorem 3:

$u_n \rightarrow u$, solution of (3.1), in weak U_1, and $u\epsilon L^\infty(\Omega)$ $(u\epsilon \ell^\infty(\Omega)$, for the finite difference problem).

REFERENCES

[1] J.S. Baras and A.J. Dorsey, "Stochastic Control of Two Partially Observed Competing Queues" to appear IEEE Trans. on Aut. Control, Oct. 1981, (Bellman special issue).

[2] D.R. Cox and W.L. Smith, Queues, John Wiley and Sons, N.Y., 1961.

[3] V.V. Rykov and E. Ye. Lembert, "Optimal Dynamic Priorities in Single Queueing Systems", Theory of Probability and its Applications, pp. 21-30.

[4] T.B. Grabill, D. Gross and M.J. Magazine, "A Classified Bibliography of Research on Optimal Design and Control of Queues", Operations Research, Vol. 25, No. 2, March-April 1977, pp. 219-232.

[5] M.J. Sobel, "Optimal Operation of Queues", in Mathematical Methods in Queueing Theory. (A.B. Clarke, Edt.) Lecture Notes in Economics and Math. Systems, 98, 1974, pp. 231-261.

[6] Z. Rosberg, P. Varaiya and J. Walrand, "Optimal Control of Service in Tandem Queues", Memorandum No. UCB/ERL M80/42, Univ. of California, Berkeley, Sept. 1980.

[7] D.L. Snyder, Random Point Processes, John Wiley, 1975.

[8] S.A. Lippman, "Semi-Markov Decision Processes with Unbounded Rewards" Management Science, Vol. 19, No. 7, 1973, pp. 717-731.

[9] H. Kushner, Introduction to Stochastic Control, Holt, Rinehart and Winston, 1971.

[10] S.A. Belbas, "Weak solutions of the Hamilton-Jacobi-Bellman equation", to appear in "Systems and Control Letters".

[11] S.A. Belbas, "Degenerate variational and quasi-variational inequalities and Bellman equations", preprint, 1981.

[12] F. Mignot and J.-P. Puel, "Inequations variationnelles et quasi-variationnelles hyperboliques du premier ordre", J. Math. Pures Appl. 55 (1976), pp. 353-378.

[13] J.L. Menaldi, Thesis: "Sur les problemes de temps d'arret, controle impulsionnel et continu correspondant a des operateurs degeneres", Paris, 1980.

Optimal Control of a M/M/1 Queue

Aurel A. Lazar

Department of Electrical Engineering
Columbia University

Abstract

The problem of optimal control of a M/M/1 queueing system is investigated. It is shown that the control maximizing the throughput under a bounded time delay criterion is a window type mechanism. The window size can be easily derived from the maximum allowed average time delay. The dependence of the maximum throughput on the time delay is also analyzed.

I. Introduction

The throughput time delay trade-off as a performance criterion for optimal control of simple queueing systems (that are used as network models) has been recently studied. In investigating this trade-off Kleinrock [3] and Barath-Kummar [1] have chosen the "power" as an optimality criterion. This criterion, however, does not display the direct dependence between the maximum throughput and the average time delay. Partly to overcome this, Schwartz [7] suggested the problem of finding a state dependent control to a queueing system with an exponential server. The realizable control should maximize the throughput under a maximum constraint on the average time delay.

In this paper an optimality criterion is introduced which displays the direct dependence between the throughput and the average time delay. The resulting optimal control maximizes the throughput under the condition that the average time delay does not exceed a preassigned value T. The offered load (or equivalently the maximum input capacity) in this optimization criterion is set to be a fixed parameter. Our investigations have shown that the optimal control of a M/M/1 system is a "window" flow control type mechanism. The window size L can be easily derived from the maximum allowed average time delay T.

The results obtained in the M/M/1 case suggest that our present views on the congestion control of computer communication networks need additional clarifications. It is a widely accepted in the computer communication networks literature that the total throughput of a network increases with the offered load from zero to a peak value and then decreases sharply to a low value (see, for example, [4] and the references therein). This rather qualitative assesment implies the necessity of flow control algorithms in computer networks. In finding the maximum atainable throughput by employing a flow control mechanism, however, not only the offered load but also the maximum time delay has to be taken into consideration. The main contribution of this paper is to show for the simple case of a M/M/1 system in what way the maximum obtainabale throughput depends on the offered load and the average time delay.

This paper is organized as follows. In section II the optimization criterion is defined. It is shown that maximizing the throughput under a maximum time delay criterion is equivalent in the M/M/1 case with minimizing the probability that the system is empty. In section III the maximum time delay that can be achieved with L packets (maximum buffer size) in the M/M/1 system is found. It is shown that the maximum achievable time delay is strictly increasing with the number of packets. The optimum control that maximizes the throughput with respect to a bounded time delay criterion is given in *Theorem* 1. The maximum obtainable throughput is also specified as a function of the existing time delay in the system. It is further shown that the so called throughput time delay function is continuous monotone increasing and piecewise convex.

II. The Optimization Criterion

Let us consider the queueing system depicted in Fig. 1. The upper queue which has an exponential server (μ_k), $1 \le k \le N$ has to be controled corresponding to a suitable optimality criterion. The lower or feedback queue has an exponential server (λ_k), $1 \le k \le N$. Since there are a maximum of N packets in the above system, the upper queue can be seen as being a M/M/1 queueing system with a finite buffer size N. Without any loss of generality, therefore, the feedback queue models the input stream to a M/M/1 system with a finite buffer size. The above setting represents, *e.g.*, a simple model of a virtual circuit in a computer network were each packet is acknowledged upon its arrival.

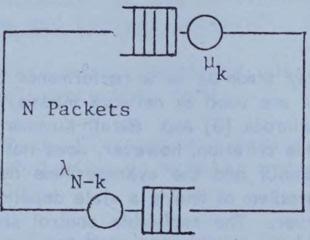


Fig.1 Model for the state dependent control of a M/M/1 queue.

The average throughput and the average time delay of the M/M/1 system (upper queue) are given by [2], [6]:

$$E\,\gamma^{(N)} = \sum_{k=1}^{N} \mu_k p_k$$

and

$$E\,\tau^{(N)} = \frac{\sum_{k=1}^{N} k p_k}{\sum_{k=1}^{N} \mu_k p_k} ,$$

respectively, where

$$p_k = \prod_{l=0}^{k-1} \frac{\lambda_{N-l}}{\mu_{l+1}} p_0$$

and

$$p_0 = \frac{1}{1 + \sum_{k=1}^{N} \prod_{l=0}^{k-1} \frac{\lambda_{N-l}}{\mu_{l+1}}}$$

denote the probabilities that the upper queue contains k packets, $1 \le k \le N$.

Definition 1. $\lambda = (\lambda_k)$, $1 \le k \le N$, will hereafter denote the control.

Folowing [5] we formaly introduce now the class of admissible controls.

Definition 2. The class of controls $\lambda = (\lambda_k)$, $1 \le k \le N$, satisfying the peak constraint

$$0 \le \lambda_k \le c \quad ,$$

for all k, $1 \le k \le N$, where c, $c \in R_+$, is a constant is called admissible.

Remark. In the language of computer communication networks the above condition implies that the access line to the upper queue has a maximum available capacity c (packets/s).

Definition 3. The control $\lambda = (\lambda_k)$, $1 \le k \le N$, is said to be optimum over the class of admissible controls for a given T, $T \in R_+$, if the maximum

$$\max_{E\tau^{(N)} \le T} E\gamma^{(N)}$$

is achieved.

Let x_k denote the expression

$$x_k = \prod_{l=0}^{k-1} \frac{\lambda_{N-l}}{\mu_{l+1}} \quad \left(= \frac{p_k}{p_0}\right) \quad ,$$

for all k, $1 \le k \le N$. Thus

$$\sum_{k=1}^{N} x_k = \sum_{k=1}^{N} \frac{p_k}{p_0} = \frac{1 - p_0}{p_0} = \frac{1}{p_0} - 1$$

and

$$p_0 = \frac{1}{1 + \sum_{k=1}^{N} x_k}.$$

In what follows we will restrict our study to the case where the M/M/1 system has a constant exponential service rate μ, *i.e.*, $\mu_k = \mu$, for all k, $1 \le k \le N$.

Intuitively, maximizing the throughput by using a bounded time delay criterion is the same as minimizing the probability that the system is empty. This assertion is proven in the following:

Lemma 1. *$\lambda=(\lambda_k)$, $1\le k\le N$, is optimum in the class of admissible controls for a given T, $T\in R_+$, if it achieves the maximum*

$$\max_{\sum_{k=1}^{N}(k-\mu T)x_k\le 0}\ \{\sum_{k=1}^{N} x_k = \frac{1}{p_0} - 1\}\ , \tag{1}$$

where $0\le x_k\le (\frac{c}{\mu})^k=\rho^k$, for all k, $1\le k\le N$.

Proof: Since the condition $E\tau^{(N)}\le T$ is equivalent with

$$\sum_{k=1}^{N}(k-\mu T)x_k\le 0$$

and

$$E\gamma^{(N)} = p_0\cdot\sum_{k=1}^{N}\mu x_k = \mu(1-\frac{1}{1+\sum_{k=1}^{N}x_k}) = \mu\cdot(1-p_0)$$

the optimum control $\lambda=(\lambda_k)$, $1\le k\le N$, satisfies (1). □

Definition 4. The mapping $F{:}R_+\to R_+$ given by

$$F(T) = \max_{E\tau^{(N)}\le T} E\gamma^{(N)}$$

is called the throughput time delay function.

Remark. The existence of F is implicitly demonstrated in section III.

III. Optimal Control of the M/M/1 Queue

In order to simplify the proof of the main theorem, a series of neccessary results will be first presented. In *Lemma* 2 we derive the expression for the maximum time delay and an achievable upper bound to it. In *Lemma* 3 it is shown that the maximum achievable time delay in a M/M/1 system is increasing with the number of packets.

Lemma 2. *The maximum average time delay $T_{\max}^{(L)}$ achieved with L packets in the system is*

$$T_{\max}^{(L)} = \frac{1}{\mu}\cdot\frac{\sum_{k=1}^{L}k\rho^k}{\sum_{k=1}^{L}\rho^k} \le \frac{L}{\mu}. \tag{2}$$

Proof:

Let us first show that

$$\frac{\sum_{k=1}^{L} kx_k}{\sum_{k=1}^{L} x_k} \le \frac{\sum_{k=1}^{L} k\rho^k}{\sum_{k=1}^{L} \rho^k} ,$$

or

$$\sum_{k=1}^{L}\sum_{k=1}^{L} x_k\rho^l(k-l) \le 0 .$$

Due to the symmetry in k and l and the fact that

$$\frac{x_k}{x_l} \le \rho^{k-l} ,$$

for all $k>l$, the expression on the left hand side is negative.

Note also that

$$T_{\max}^{(L)} < \frac{L}{\mu}$$

since

$$\sum_{k=1}^{L} (L-k)\rho^k > 0 . \qquad \square$$

Remark. The right hand side of inequality (2) can be obtained when $\rho\to\infty$. This corresponds to the situation when the served packets rejoin instanteneously the queue. L/μ is then exactly the average time delay of a M/M/1 queue having constantly L packets in the system. A simple equivalent expression for $T_{\max}^{(L)}$ is derived in the *Corollary* to the *Theorem* 1.

Lemma 3. *The maximum average time delay in the M/M/1 system is strictly increasing with the number of packets L, i.e.,*

$$T_{\max}^{(L)} < T_{\max}^{(L+1)} ,$$

for all L, L∈N.

Proof: We have to show that

$$\frac{\sum_{k=1}^{L} k\rho^k}{\sum_{k=1}^{L} \rho^k} < \frac{\sum_{k=1}^{L+1} k\rho^k}{\sum_{k=1}^{L+1} k\rho^k} ,$$

or

$$\sum_{k=1}^{L} (L+1-k)\rho^{k} > 0 \ .$$

The latter inequality is true since $k < L+1$. □

We can give now the following:

Theorem 1. *Given that $T_{\max}^{(L-1)} < T \le T_{\max}^{(L)}$, $2 \le L \le N$, the optimum control of a M/M/1 queue with a maximum of N packets in the system is given by*

$$\lambda_k = \begin{cases} c & N-L+2 \le k \le N \\ \dfrac{1}{L/\mu - T} \displaystyle\sum_{l=1}^{L-1} (\mu T - l)\rho^{l+1-L} & k = N-L+1 \\ 0 & 1 \le k \le N-L \end{cases} \ .$$

Finally, the maximum throughput amounts to

$$F(T) = \mu\Bigl(1 - \frac{1}{1 + \dfrac{1}{L-\mu T} \displaystyle\sum_{k=1}^{L-1} (L-k)\rho^{k}}\Bigr) \ .$$

Proof:

Let us assume that

$$T_{\max}^{(L-1)} < T \le T_{\max}^{(L)} \ ,$$

where $2 \le L \le N$. The condition

$$\sum_{k=1}^{N} (k - \mu T)x_k \le 0,$$

can be written as:

$$(L - \mu T)x_L \le -(N - \mu T)x_N - \cdots - [(L+1) - \mu T]x_{L+1}$$
$$- [(L-1) - \mu T]x_{L-1} - \cdots - (2 - \mu T)x_2 - (1 - \mu T)x_1 \ ,$$

or since $L - \mu T > 0$

$$x_L \le -\frac{N-\mu T}{L-\mu T}x_N - \cdots - \frac{L+1-\mu T}{L-\mu T}x_{L+1}$$
$$- \frac{L-1-\mu T}{L-\mu T}x_{L-1} - \cdots - \frac{2-\mu T}{L-\mu T}x_2 - \frac{1-\mu T}{L-\mu T}x_1 \ .$$

Thus

$$\sum_{k=1}^{N} x_k \le -\frac{N-L}{L-\mu T}x_N - \cdots - \frac{1}{L-\mu T}x_{L+1} + \frac{1}{L-\mu T}x_{L-1}$$

$$+ \cdots + \frac{L-2}{L-\mu T} x_2 + \frac{L-1}{L-\mu T} x_1$$

$$\le \frac{1}{L-\mu T} \sum_{k=1}^{L-1} (L-k)\rho^k \ .$$

Equality can be achieved iff

$$x_k = \begin{cases} 0 & L+1 \le k \le N \\ \dfrac{1}{L-\mu T} \displaystyle\sum_{l=1}^{L-1} (\mu T - l)\rho^l & k = L \\ \rho^k & 1 \le k \le L-1 \end{cases} \ .$$

It remains only to show that

$$0 \le x_L \le \rho^L \ ,$$

or

$$0 \le \sum_{k=1}^{L-1} (\mu T - k)\rho^k \le (L - \mu T)\rho^L \ ,$$

which is equivalent to

$$\frac{1}{\mu} \cdot \frac{\displaystyle\sum_{k=1}^{L-1} k\rho^k}{\displaystyle\sum_{k=1}^{L-1} \rho^k} \le T \le \frac{1}{\mu} \cdot \frac{\displaystyle\sum_{k=1}^{L} k\rho^k}{\displaystyle\sum_{k=1}^{L} \rho^k} \ ,$$

or

$$T_{\max}^{(L-1)} \le T \le T_{\max}^{(L)} \ . \qquad \square$$

Remark. Since the lower queue will contain at all times at least $N-L$ packets, the optimum control can also be achieved by a control scheme using a total of $N=L$ packets. We require that

$$\lambda_k = \begin{cases} c & 2 \le k \le L \\ \dfrac{1}{L/\mu - T} \displaystyle\sum_{l=1}^{L-1} (\mu T - l)\rho^{l+1-L} & k = 1 \end{cases} \ .$$

Therefore the optimal control is a window type mechanism. The window size L can be easily derived from the maximum time delay of the system. Naturally the number of packets will depend on the maximum offered load (or line capacity into the system).

Corollary. If $T = T_{\max}^{(L)}$ *then*

$$F(T^{(L)}_{\max}) = c \cdot \frac{1 - \rho^{L}}{1 - \rho^{L+1}}$$

and

$$T^{(L)}_{\max} = \frac{1}{\mu}\left(\frac{1}{1-\rho} + \frac{L}{1-\rho^{-L}}\right) .$$

Proof: From $T = T^{(L)}_{\max}$ we have

$$\sum_{k=1}^{L-1} (\mu T - k)\rho^{k} = (L - \mu T)\rho^{L}$$

and therefore

$$\lambda_1 = \frac{\mu}{L - \mu T}\rho^{1-L}\sum_{k=1}^{L-1}(\mu T - k)\rho^{k} = c \quad .$$

The corresponding throughput amounts to

$$F(T^{(L)}_{\max}) = \mu\left(1 - \frac{1}{1 + \sum_{k=1}^{L} \rho^{k}}\right) ,$$

or

$$F(T^{(L)}_{\max}) = c \cdot \frac{1 - \rho^{L}}{1 - \rho^{L+1}} \quad .$$

The computation of the time delay can be similarly done. □

Remark. On can easily see that

$$\lim_{L\to\infty} F(T^{(L)}_{\max}) = \begin{cases} c & \rho < 1 \\ \mu & \rho \geq 1 \end{cases}$$

and

$$\lim_{L\to\infty} T^{(L)}_{\max} = \begin{cases} \dfrac{1}{\mu - c} & \rho < 1 \\ \infty & \rho \geq 1 \end{cases} .$$

Lemma 4. *The throughput time delay function is continuous nondecreasing on R_+. Furthermore, for all T, $T^{(L-1)}_{\max} \leq T \leq T^{(L)}_{\max}$, $L \in N$, F is convex.*

Proof: The proof can be easily supplied. □

For $\mu=10$ and $c=7$ the throughput time delay function is depicted in Fig. 2. Fig. 3 depicts the parametrized throughput time delay function for various values of c. The proof of the "overall concave" behavior of the throughput time delay function is left to the reader as a straightforward exercise.

IV. Conclusion

In this paper we have shown that the control that maximizes the throughput of a M/M/1 system under a bounded time delay criterion is state dependent. The number of packets that, when optimaly controled lead to the maximum throughput has been shown to be easily derivable from the preassigned upper bound of the average time delay. The dependence of the maximum throughput on the average time delay has also been investigated. The analysis of the M/M/1 system has shown that it is appropiate to discuss congestion problems in terms of maximum throughput under a bounded time delay criterion. The offered load then appears only as a parameter.

Acknowledgement

The author would like to thank Prof. Mischa Schwartz for helpful discussions on this and related topics. The research reported here was supported by the National Science Foundation under grant # ECS 8106476.

References

[1] Bharath-Kummar, K., "Optimum End-to-End Flow Control in Networks," *Conference Record of the International Conference on Communications,* pp. 23.3.1 - 32.3.6, June 1980.

[2] Kleinrock, L., *Queueing Systems, Volume I: Theory,* John Wiley, New York, 1975.

[3] Kleinrock, L., "Power and Deterministic Rules of Thumb for Probabilistic Problems in Computer Communications," *Conference Record of the International Conference on Communications,* pp. 43.1.1 - 43.1.10, June 1979.

[4] Lam, S.S., and Reiser, M., "Congestion Control of Store-and-Forward Networks by Input Buffer Limits - An Analysis", *Trans. on Communications,* Vol. **Com-27,** No. 1. January 1979.

[5] Lazar, A.A., *Optimal Information Processing Using Counting Point Process Observations,* Ph.D. Dissertation, Dept. of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ 08544, Oct. 1980.

[6] Schwartz, M., *Computer Communication Networks Design and Analysis,* Prentice Hall, Englewood Cliffs, 1976.

[7] Schwartz, M., Routing and Flow Control in Data Networks, *IBM Research Report,* RC **8353 (#36319),** IBM T.J. Watson Research Center, Yorktown Heights, 1980.

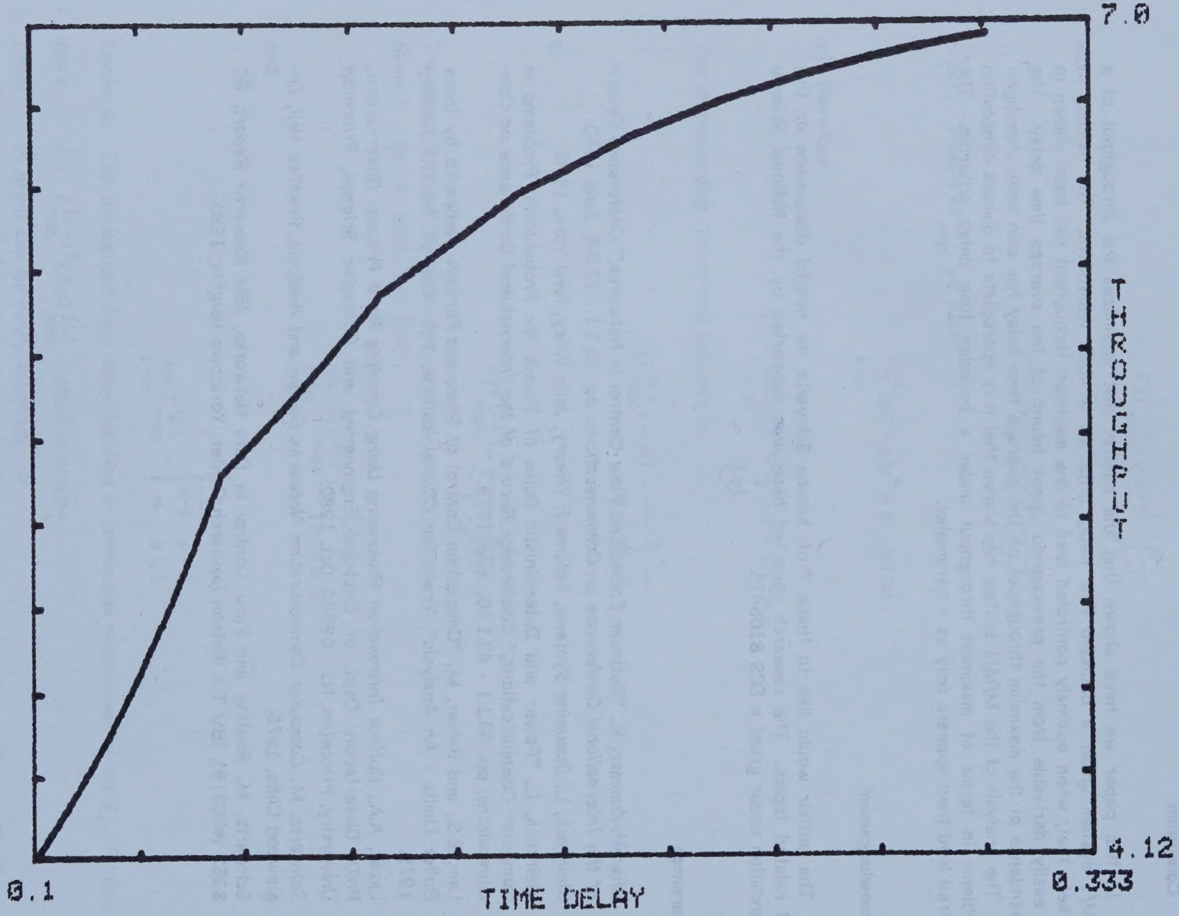


Fig. 2 The throughput time delay function ($\mu = 10$, $c = 7$).

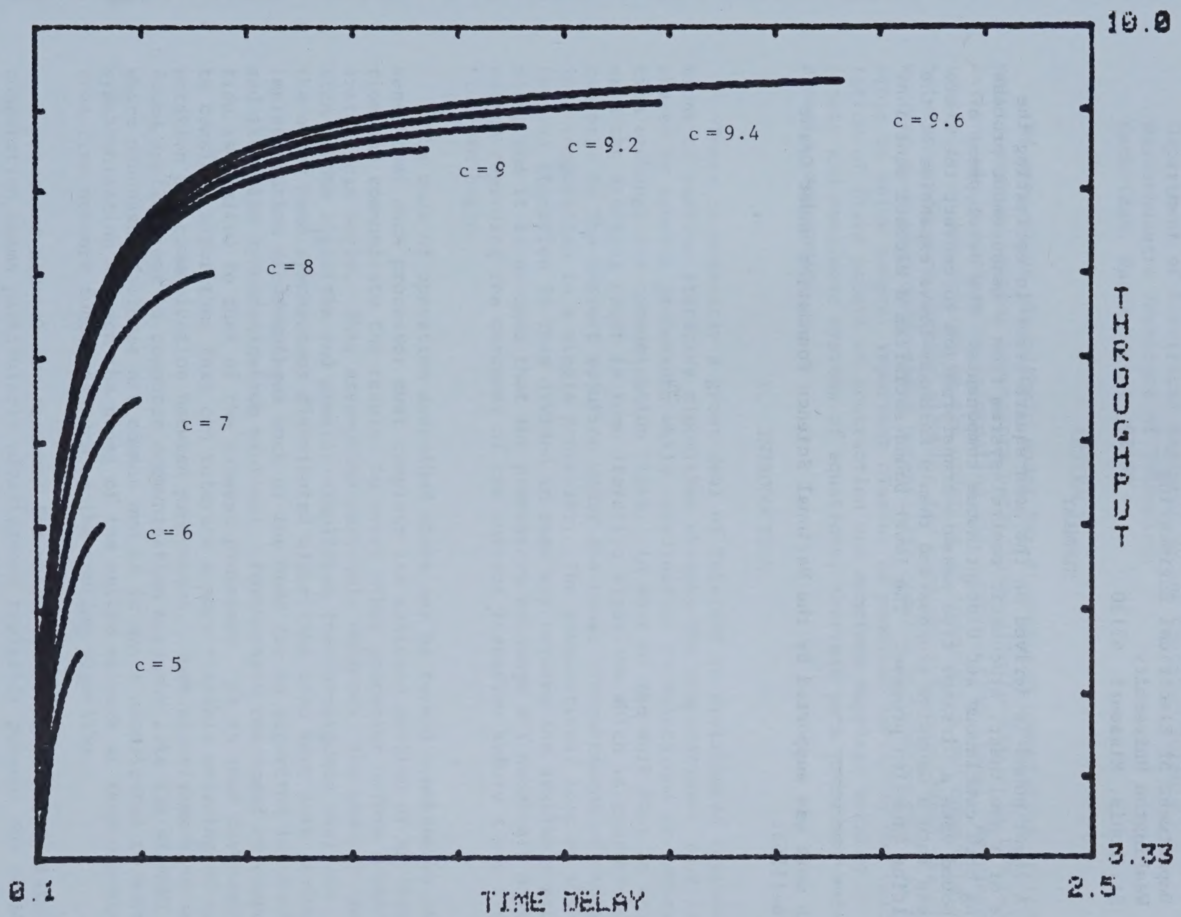


Fig. 3 The dependence of the throughput time delay function on the parameter c. (μ = 10)

A CRAMER-RAO TYPE LOWER BOUND ON ESTIMATION PERFORMANCE FOR STOCHASTIC CONTROL-SYSTEMS*

CHENG-WU CHEN and DONALD L. SNYDER
Department of Electrical Engineering
Washington University
St. Louis, Missouri 63130

SUMMARY

A lower bound is derived on the mean square-error in estimating the state of a nonlinear, stochastic control-system from a measurement process having both continuous and discontinuous components. The development of the bound uses a Girsanov type measure-transformation to convert the state process into a Gaussian process and then a Karhunen-Loeve expansion of the resulting Gaussian process. The lower bound satisfies a Riccati equation.

---

*This work was supported by the National Science Foundation under Grant ENG76-11565.

A MODEL FOR UNCOORDINATED DISTRIBUTED COMPUTATION OF FIXED POINTS*

DIMITRI P. BERTSEKAS
Laboratory for Information and Decision Systems
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

ABSTRACT

We present an algorithmic model for distributed computation of fixed points whereby several processors participate simultaneously in the calculations while exchanging information via communication links. We place essentially no assumptions on the ordering of computation and communication between processors thereby allowing for completely uncoordinated execution. We find that even under these potentially chaotic circumstances it is possible to solve several important classes of problems including the calculation of fixed points of contraction and monotone mappings arising in linear and nonlinear systems of equations, shortest path problems, and dynamic programming.

1. INTRODUCTION

There is presently a great deal of interest in distributed implementations of various iterative algorithms whereby the computational load is shared by several processors while coordination is maintained by information exchange via communication links. In most of the work done in this area the starting point is some iterative algorithm which is guaranteed to converge to the correct solution under the usual circumstances of centralized computation in a single processor. The computational load of the typical iteration is then divided in some way between the available processors, and it is assumed that the processors exchange all necessary information regarding the outcomes of the current iteration before a new iteration can begin.

The mode of operation described above may be termed synchronous in the sense that each processor must complete its assigned portion of an iteration and communicate the results to every other processor before a new iteration can begin. This assumption certainly enhances the orderly operation of the algorithm and greatly simplifies the convergence analysis. On the other hand synchronous distributed algorithms also have some obvious implementation disadvantages such as the need for an algorithm initiation and iteration synchronization protocol. Furthermore the speed of computation is limited to that of the slowest processor. It is thus interesting to consider algorithms that can tolerate a more flexible ordering of computation and communication between processors. Such algorithms have so far found applications in computer communication networks like the ARPANET [1] where processor failures are common and it is quite complicated to maintain synchronization between the nodes of the entire network as they execute real-time network functions such as the routing algorithm.

Processor network environments for which weakly coordinated distributed computation seems particularly advantageous typically possess one or more

---

*This research was conducted at the M.I.T. Laboratory for Information and Decision Systems with partial support provided by the Defense Advanced Projects Agency under Contract No. ONR-N00014-75-C-1183.

of the following characteristics all of which involve occurance of some type of unpredictable event.

1) Computation nodes and communication links are subject to frequent and/or unexpected failures. (For example packet radio networks).

2) Computation nodes have different and/or time varying speeds of execution. (For example each processor is assigned to a perhaps time varying number of tasks involving computation loads which are not fixed a priori).

3) Computation at various nodes is event driven. (For example in data collection or sensor networks where the timing, and ordering of measurements may not be predictable).

It is possible to consider various degrees of coordination in different types of distributed algorithms. An interesting question is to determine the minimum degree of coordination needed in a given algorithm in order to obtain the correct solution. To this end we consider an extreme model of uncoordinated distributed algorithms whereby computation and communication are performed at each processor completely independently of the progress in other processors. It is perhaps surprising that even under these chaotic circumstances it is still possible to solve correctly important classes of fixed point problems. The complete analysis is given in Ref. [2] for broad classes of dynamic programming and in Ref. [3] for more general fixed point problems involving contraction and monotonicity assumptions. In the present paper we describe the algorithmic model and indicate the type of results that can be shown in some generality.

2. A MODEL FOR DISTRIBUTED UNCOORDINATED FIXED POINT ALGORITHMS

The fixed point problem considered in this paper is defined in terms of a set X, a class F of functions mapping X into the extended real line $[-\infty,+\infty]$, and a mapping T which maps F into itself. We wish to find an element J^* of F such that

$$J^* = T(J^*). \tag{1}$$

or equivalently

$$J^*(x) = T(J^*)(x), \qquad \forall x \in X, \tag{2}$$

where $J^*(x)$ and $T(J^*)(x)$ denote the values of the functions J^* and $T(J^*)$ respectively at the typical element $x \in X$. We will assume throughout that T has a unique fixed point J^* within the set F.

We provide some examples:

Example 1: (Fixed points of mappings on R^n). Let X be the finite set

$$X = \{1,2,\ldots,n\},$$

and F be the set of all real-valued functions on X. Then F can be identified with the n-dimensional space R^n in the sense that with each $J \in F$ we can associate the n-dimensional vector $J(1),J(2),\ldots,J(n)$. Similarly $T(J)$ can be identified with the n-dimensional vector $T(J)(1),\ldots,T(J)(n)$, so the fixed point problem (1) amounts to solving the system of n equations

$$J^* = T(J^*) \quad \text{or} \quad J^*(i) = T(J^*)(i), \qquad \forall i = 1,\ldots,n \tag{3}$$

with the n unknowns $J^*(1),\dots,J^*(n)$. It is also evident that any system of n (possibly nonlinear) equations with n unknowns can be formulated into a fixed point problem such as (3).

Example 2: (Shortest path problems). Let (N,L) be a directed graph where $N = \{1,2,\dots,n\}$ denotes the set of nodes and L denotes the set of links. Let $N(i)$ denote the downstream neighbors of node i, i.e., the set of nodes j for which (i,j) is a link. Assume that each link (i,j) is assigned a positive scalar a_{ij} referred to as its length. Assume also that there is a directed path to node 1 from every other node. Then it is known ([2], p.67) that the shortest path distances $J^*(i)$ to node 1 from all other nodes i solve uniquely the equations

$$J^*(i) = \min_{j\varepsilon N(i)} \{a_{ij} + J^*(j)\} , \qquad i \neq 1 \tag{4a}$$

$$J^*(1) = 0 \tag{4b}$$

If we make the identifications $X = \{1,2,\dots,n\}$, F: Set of all functions mapping X into $[0,+\infty]$, and define $T(J)$ for all $J\varepsilon F$ by means of

$$T(J)(i) = \begin{cases} \min_{j\varepsilon N(i)} \{a_{ij} + J(j)\} & \text{if } i \neq 1 \\ 0 & \text{if } i = 1 \end{cases} \tag{5}$$

then we find that the fixed point problem (2) reduces to the shortest path problem.

The shortest path problem above is representative of a broad class of dynamic programming problems which can be viewed as special cases of the fixed point problem (2) and can be correctly solved by using the distributed algorithms of this paper (see [3]).

Our algorithmic model can be described in terms of a collection of n computation centers (or processors) referred to as nodes and denoted 1,2,...,n. The set X is partitioned into n disjoint sets denoted $X_1,\dots, X_n$, i.e.

$$X = \bigcup_{i=1}^{n} X_i \ , \quad X_i \cap X_j = \emptyset, \quad \text{if } i \neq j.$$

Each node i is assigned the responsibility of computing the values of the solution function J^* [c.f. (1),(2)] at all $x\varepsilon X_i$.

At each time instant, node i can be in one of three possible states compute, transmit, or idle. In the compute state node i computes a new estimate of the values of the solution function J^* for all $x\varepsilon X_i$. In the transmit state node i communicates the estimate obtained from the latest computation to one or more nodes j ($j\neq i$). In the idle state node i does nothing related to the solution of the problem. It is assumed that a node can receive a transmission from other nodes simultaneously with computing or transmitting, but this is not a real restriction since, if needed, a time period in a separate receive state can be lumped into a time period in the idle state.

We assume that computation and transmission for each node takes place in uninterupted time intervals $[t_1,t_2]$ with $t_1 < t_2$, but do not exclude the possibility that a node may be simultaneously transmitting to more than one nodes nor do we assume that the transmission invervals to these nodes have the same origin and/or termination. We also make no assumptions on the length, timing and sequencing of computation and transmission intervals other than the following:

Assumption (A): There exists a positive scalar P such that, for every node i, every time interval of length P contains at least one computation interval for i and at least one transmission interval from i to each node $j \neq i$.

Each node i also has a buffer B_{ij} for each $j \neq i$ where it stores the latest transmission from j, as well as a buffer B_{ii} where it stores its own estimate of values of the solution function for all $x \in X_i$. The contents of each buffer B_{ij} at time t are denoted J^t_{ij}. Thus J^t_{ij} is, for every t, a function from X_j into $[-\infty,\infty]$ and may be viewed as the estimate by node i of the restriction of the solution function J* on X_j available at time t. The rules according to which the functions J^t_{ij} are updated are as follows:

1) If $[t_1,t_2]$ is a transmission interval from node j to node i the contents $J^{t_1}_{jj}$ of the buffer B_{jj} at time t_1 are transmitted and entered in the buffer B_{ij} at time t_2, i.e.

$$J^{t_2}_{ij} = J^{t_1}_{jj}. \tag{6}$$

2) If $[t_1,t_2]$ is a computation interval for node i the contents of buffer B_{ii} at time t_2 are replaced by the restriction of the function $T(J^{t_1}_1)$ on X_i where, for all t, J^t_i is defined by

$$J^t_i(x) = \begin{cases} J^t_{ii}(x) & \text{if } x \in X_i \\ J^t_{ij}(x) & \text{if } x \in X_j, \quad j \neq i \end{cases} \tag{7}$$

In other words we have

$$J^{t_2}_{ii}(x) = T(J^{t_1}_i)(x), \quad \forall\, x \in X_i \tag{8}$$

3) The contents of a buffer B_{ii} can change only at the end of a computation interval for node i. The contents of a buffer B_{ij}, $j \neq i$ can change only at the end of a transmission interval from j to i.

Additional conditions under which there holds

$$\lim_{t \to \infty} J_i^t(x) = J^*(x), \quad \forall x \varepsilon X_i, \; i = 1, \ldots, n \tag{9}$$

may be found in [2],[3]. An interesting aspect of results of this type is that they do not require that the initial processor buffer contents be identical and indeed these initial conditions can vary within a broad range. This means that for problems that are being solved continuously in real time it is not necessary to reset the initial conditions and resynchronize the algorithm each time the problem data changes. As a result the potential for tracking slow variations on the solution function is improved and algorithms implementation is considerably simplified.

REFERENCES

[1] J. McQuillan, G. Falk, and I. Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm", IEEE Trans. on Communications, Vol. COM-26, 1978, pp. 1802-1811.

[2] D. P. Bertsekas, "Distributed Dynamic Programming", Report LIDS-P-1060, Laboratory for Information and Decision Systems, Mass. Institute of Technology, Cambridge, Mass., Dec. 1980 (to appear in IEEE Trans. on Automatic Control).

[3] D. P. Bertsekas, "Distributed Computation of Fixed Points", Report LIDS-P-1135, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Mass., Aug. 1981.

[4] E. L. Lawler, Combinatorial Optimization: Networks and Matroids, Holt, Rinehart, and Winston, N.Y., 1976.

FEEDBACK STABILIZATION AND CONTROL OF LINEAR NEUTRAL SYSTEMS

T. J. TARN
Washington University
St. Louis, MO 63130

M. W. SPONG
Lehigh University
Bethlehem, PA 18015

K. ITO
NASA Langley Center
Hampton, VA 23665

ABSTRACT

New results for linear neutral systems are reported in this paper. Specifically we obtained the following: (i) realization and feedback stabilization of linear neutral systems with discrete delays using ring theoretic approach, (ii) feedback synthesis of a fairly general class of linear neutral systems via functional analytic approach.

1. INTRODUCTION

The theory of linear dynamical systems defined over commutative rings has been studied with increasing vigor over the past several years. One of the primary motivations for studying systems over rings has been the application to retarded delay-differential equations, which have themselves been extensively studied and applied to many areas of engineering and applied mathematics, including the modeling of epidemics, [1], chemical reactors [2] and transmission lines [3]. The study of transmission lines is particularly noteworthy, for in this case one is often led to consider neutral delay-differential equations, i.e. equations with delay in the (highest order) derivative. For example, Brayton [4] and Brayton and Miranker [5] considered networks of lumped elements interconnected by transmission lines and showed that the mixed initial-boundary value problems that arise can give rise to neutral delay-differential equations. More generally, Cooke and Krumme [6] presented a general method for reducing a large class of initial-boundary value problems for hyperbolic PDE's to initial-value problems for neutral differential-difference equations.

In this paper the first problem we investigate is that of realization and stabilization of linear neutral systems with discrete delays. We show that any autonomous, linear neutral system with discrete delays is zero-state equivalent to an abstract linear system over a local ring of operators.

Using the abstract model we settle the basic existence question for neutral realizations. For general infinite dimensional linear systems, the precise analog of the finite dimensional state space isomorphism theorem is lacking. A consequence of this is the necessity of introducing the notion of spectral minimality. In the case of single input-single output systems we obtain realizations which are both minimal and spectrally minimal. In the multi-variable case we may still obtain minimal realizations under quite general conditions but only at the expense of spectral minimality. Using the Cruz-Hale theory of stable D-operators we give conditions that ensure that any poles introduced into the realization are strictly contained in the left half plane and indeed are characterized as characteristic values of the D-operator.

Next, the problem of feedback stabilization of neutral systems is studied using the abstract model. We show that, for neutral systems with commensurable delays and stable D-operator in the sense of Cruz and Hale, Morse's Theorem [7] on pole-assignment over a PID implies stabilizability in the reachable case.

For systems with arbitrary D-operators, derivative feedback may be used to reduce the problem to the case of stable D-operators.

The second problem considered in this paper is the optimal control of linear neutral systems. Recently there has been a considerable amount of

---

This research was supported in part by the National Science Foundation under Grant Nos. ECS-8017184 and INT-7902976.

research concerning abstract linear control systems modeled by evolution equation in a Hilbert Space X,

$$\frac{d}{dt} z(t) = Az(t) + Bu(t) \quad , \quad z(0) \in X \, , \tag{1}$$

where A is the infinitesimal generator of a strongly continuous semigroup of linear bounded operators on X and B is a linear bounded operator mapping from the control space U (a Hilbert space) into X. [8], [9]. As is known, the linear retarded control systems are modeled by (1), taking X to be the Product space $\mathbb{R}^n \times L_2$. By using such models the controllability problem [10] and the optimal control problem [11] for linear retarded systems may be treated in mathematically unified manner.

Consider the linear neutral FDE in $\mathbb{R}^n$

$$\frac{d}{dt} Dx_t = Lx_t + Bu(t) \quad , \quad x_0 = \psi. \tag{2}$$

The control function u(t) takes values in $\mathbb{R}^m$. Take B to be an (nxm) matrix, D, L are the linear operators: $M([-r,0]:\mathbb{R}^n)$, the measurable functions on $[-r,0]$ into $\mathbb{R}^n \to \mathbb{R}^n$ which are continuous when restricted to $C([-r,0]:\mathbb{R}^n)$, and D has no singular part. Note that (2) may be modeled by an evolution equation in $\mathbb{R}^n \times L_2$,

$$\frac{d}{dt} z(t) = \tilde{A}z(t) + \tilde{B}u(t) \quad , \quad z(0) = (\psi^0, \psi^1), \tag{3}$$

where $\tilde{A}$ is the infinitesimal generator, $\tilde{B} : \mathbb{R}^n \to \mathbb{R}^n \times L_2$ is defined by $\tilde{B}u = (Bu, 0) \in \mathbb{R}^n \times L_2$ for $u \in \mathbb{R}^m$ and $z(t) \overset{\text{def}}{=} (Dx_t, x_t)$.

Regarding the optimal control problem of linear neutral systems with the quadratic cost;

$$\int_0^T (|Q_0 x(t)|^2 + |u(t)|^2) dt \quad , \quad \text{for some } T > 0 \, , \tag{4}$$

where Q_0 is a matrix of dimension (pxn), the existence of optimal controls is discussed in [12], [13], choosing the initial function in C. What seems to be a more difficult problem is to synthesize the optimal feedback controller.

For a single point delay case, Datko [14] has shown that the optimal control satisfies a linear feedback law for the initial function $\psi \in C$. However, to our knowledge no explicit form for the optimal feedback law, i.e., no Riccati equation has been obtained.

Note that if $\tilde{Q}z \overset{\text{def}}{=} Q_0 z^1(0)$ for $z = (z^0, z^1) \in \mathbb{R}^n \times L_2$, then (4) is transformed into

$$\int_0^T (||Qz(t)||^2 + |u(t)|^2) dt \tag{5}$$

within the formulation (3). Hence it seems that the above control problem can be analyzed by general methods in [8], [9], applicable to (1). However it is not possible to apply them to our problem directly, since $\breve{Q}$ is not continuous as a mapping from $\mathbb{R}^n \times L_2$ into $\mathbb{R}^p$. We shall overcome this difficulty by exploiting the idea of Balakrishnan [15] who has studied the boundary control problem of parabolic equations; i.e., the original problem is treated as the "limiting problem" of a family of well-posed problems in which the operator $\breve{Q}$ is replaced by $\tilde{Q}J_n$ in (5), where $J_n = n(nI-\tilde{A})^{-1}$, $n \in \rho(\tilde{A})$. The principal result is the establishment of feedback solution in terms of the solution to a class of Riccati equations.

Our method when combined with the investigation of Zabczyk [16] is nicely extended to the infinite-time quadratic cost problem by studying the Algebraic Riccati Equation. In this discussion the detectability assumption plays a key role.

Because of lack of space, we will only present the results and omit proofs, which may be found in [17, 18].

2. REALIZATION AND STABILIZATION

We consider the class of autonomous linear neutral systems with discrete delays given by

$$\begin{aligned} \dot{x}(t) &= \sum_{k=1}^{N} A_{-k}\dot{x}(t-\tau_k) + \sum_{k=0}^{N} A_k x(t-\tau_k) + \sum_{k=0}^{M} B_k u(t-\tau_k) \\ y(t) &= \sum_{k=1}^{P} C_k x(t-\tau_k) \end{aligned} \tag{6}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^p$, A_k, B_k and C_k are constant matrices of appropriate order and $0 = \tau_0 < \tau_1 < \ldots < \tau_T$, $T = \max\{N,M,P\}$.

For any real number $\tau_j > 0$ we define the formal delay or shift operator σ_j by $(\sigma_j x)(t) := x(t-\tau_j)$. Given a finite set of real numbers $0 < \tau_1 < \ldots < \tau_T$ there is a rationally independent set $0 < h_1 < \ldots < h_k$ which generates $\{\tau_1,\ldots,\tau_T\}$, i.e. $\{h_1,\ldots,h_k\}$ is a free basis for the Z-module $M = Z\tau_1 + \ldots + Z\tau_T$. See [19]. We shall always assume therefore that the operators $\sigma_1,\ldots,\sigma_k$ correspond to rationally independent delays and we use the multi-index notation $\sigma = (\sigma_1,\ldots,\sigma_k)$ and $\mathbb{R}[\sigma] = \mathbb{R}[\sigma_1,\ldots,\sigma_k]$, the ring of polynomials in $\sigma_1,\ldots,\sigma_k$ with coefficients in $\mathbb{R}$. The 'delay action' of σ is naturally extended to the vector case by

$$\sigma_j(x_1(t),\ldots,x_n(t))^T = (\sigma_j x_1(t),\ldots,\sigma_j x_n(t))^T.$$

Let $m=(\sigma)=(\sigma_1,\ldots,\sigma_k)$ the ideal generated by $\sigma_1,\ldots,\sigma_k$. We denote by π the canonical surjection $\mathbb{R}[\sigma] \to \mathbb{R}[\sigma]/m$. π is then 'evaluation at the origin'. If Γ is an nxn matrix over $\mathbb{R}[\sigma]$ we shall use $\pi\Gamma$ to mean componentwise evaluation.

Consider now the neutral system (6). We may choose an independent set of delay operators $\sigma_1,\ldots,\sigma_k$ and write (6) as

$$D(\sigma)\dot{x}(t) = \tilde{A}(\sigma)x(t) + \tilde{B}(\sigma)u(t)$$

$$y(t) = \tilde{C}(\sigma)x(t) \tag{7}$$

where $D(\sigma)$ is atomic at zero [20], i.e., $\pi D(\sigma)$ is non-singular.

Set $S = \{p(\sigma) \in \mathbb{R}[\sigma] | \pi p \neq 0\}$. Then S is a multiplicative set. Furthermore, if $D(\sigma) \in \mathbb{R}[\sigma]^{nxn}$ is atomic at zero it follows from elementary matrix theory that $\det D(\sigma) \in S$ and hence $D^{-1}(\sigma)$ exists over the localization $R_S = S^{-1}\mathbb{R}[\sigma]$.

Given the neutral system (7) over $R = \mathbb{R}[\sigma]$ we define the associated R_S-system as

$$\begin{aligned} \dot{x}(t) &= A(\sigma)x(t) + B(\sigma)u(t) \\ y(t) &= C(\sigma)x(t) \end{aligned} \tag{8}$$

where $A(\sigma) = D^{-1}(\sigma)\tilde{A}(\sigma)$, $B(\sigma) = D^{-1}(\sigma)\tilde{B}(\sigma)$, and $C(\sigma) = \tilde{C}(\sigma)$.

Theorem 1: Let $u(t)$ be locally integrable for $\tau > 0$ and $u(t) \equiv 0$, $t < 0$. Then $x(t) \in C_0^n$ is a solution of (7) iff it is a solution of (8).

The basic tenet of finite dimensional realization theory, namely "rational <=> realizable" remains valid in the present context. This settles the basic existence question for realization of neutral systems which we state as

Theorem 2: Let $H(s,\sigma) \in \mathbb{R}(s,\sigma)^{pxm}$ be a transfer function. Then the following statements are equivalent:

1) $H(s,\sigma)$ has a neutral realization of the form (6).
2) $H(s,\sigma)$ has an abstract realization over R_S.
3) $H(s,\sigma)$ can be written as a proper rational (matrix) function over R_S.

A system Σ is said to be spectrally minimal iff the roots of the (open loop) characteristic equation of Σ coincide with the set of poles of the system transfer function. If $\Sigma = (X,D,\tilde{A},\tilde{B},\tilde{C})$ is a neutral realization of $H(s) = Q(s)/p(s)$, then Σ is spectrally minimal iff the roots of $\chi(s) = |sD-\tilde{A}|$ coincide with the poles of $H(\sigma)$.

From the proof of Theorem 2 we see that the neutral realization given by the theorem is spectrally minimal. For multiple input-multiple output systems ($m,p>1$) however the above realization will not, in general, be minimal in the sense of minimal dimension of the defining matrices. Thus to obtain a minimal realization we have had to pay a price, namely spectral minimality. However a usable realization must not introduce undesirable poles. To accomplish the goal we need the following notion.

$D(\sigma) \in \mathbb{R}[\sigma]^{nxn}$ is said to be stable or stable in the sense of Cruz and Hale iff $\det D(e^{-sh_1},\ldots,e^{-sh_k}) = 0 \Rightarrow \operatorname{Re} s \leq \beta$ for some $\beta < 0$. Let $\tilde{S} \subseteq S$ be the subset of S consisting of all such stable polynomials, then we may define $R_{\tilde{S}} = \tilde{S}^{-1}\mathbb{R}[\sigma]$.

Theorem 3: Let $H(s,\sigma) \in \mathbb{R}(s,\sigma)^{pxm}$. Then the following statements are equivalent:

1) $H(s,\sigma)$ has a neutral realization of the form (6) with stable D-operator.

2) $H(s,\sigma)$ may be written as a proper rational function over $R_{\tilde{S}}$.

3) $d_n(\sigma)$ is stable, where $H(s,\sigma) = \frac{Q(s,\sigma)}{p(s,\sigma)}$, $p(s,\sigma) = d_n(\sigma)s^n + \ldots + d_0(\sigma)$.

Theorem 4: Let $f = (A_1, A_2, \ldots)$ be a response map over $R_{\tilde{S}}$ with Hankel matrix H^f of rank n. Then the following statements are equivalent:

1) f has a split realization over $R_{\tilde{S}}$.

2) There exists an nxn submatrix Φ of H^f such that $\det\Phi \in \tilde{S}$, i.e., Φ^{-1} exists over $R_{\tilde{S}}$.

Furthermore the spectrum of the system consists of the poles of the transfer function $H(s,\sigma)$ together with the eigenvalues of the stable D-operator.

Now we consider the problem of feedback stabilization for linear neutral systems. To be specific we consider the effect of (state module) feedback on the characteristic polynomial of the equation which, as is well-known, determines the asymptotic behavior of the system.

For systems with stable D-operator we need only consider the $R_{\tilde{S}}$-associated system. For the remainder of this section we shall make the assumption that the delays appearing in the system are commensurable. In this case $R_{\tilde{S}}$ is a p.i.d. and thus we may use the well-known theorem of Morse [7] to stabilize (7).

Theorem 5: A sufficient condition for stabilizability of the neutral system (7) with stable D-operator and commensurable delays is that the $R_{\tilde{S}}$-associated system be reachable.

We next consider the neutral system (7) without the restriction that $D(\sigma)$ be stable. Suppose that there is a $k(\sigma)$ such that $\bar{D} := D + Bk$ is a stable D-operator. Then, setting

$$u(t) = -k(\sigma)\dot{x}(t) + \nu(t)$$

yields the system

$$\bar{D}(\sigma)\dot{x}(t) = \tilde{A}(\sigma)x(t) + \tilde{B}(\sigma)\nu(t)$$

thus providing a reduction to the case of a stable D-operator. Sufficient conditions on the pair $(D,\tilde{B})$ so that such a k exists are given in [17].

Systems with non-commensurable delays are much more difficult to handle in this context. Indeed, the counterexample of Bumby et. al. [21] shows that some hypotheses, e.g. scalar input, constancy of the Kronecker indices, etc., in addition to reachability are needed to achieve pole-placement using state-module feedback. By solving a certain variational problem we may still achieve stabilization in the reachable case (but no pole assignment in general). This last result is based on an idea of Byrnes and will appear in [22].

3. LQR-THEORY

In this section we consider the Linear Quadratic Regulator (LQR) problem for linear neutral systems described by (2). We will work in the product space $M_2 \tilde{=} \mathbb{R}^n \times L_2([-r,0];\mathbb{R}^n)$.

Problem: For fixed interval $[t_0,T]$ minimize over $u(\cdot)$ in $L_2([t_0,T];\mathbb{R}^m)$

$$J(t_0,T;u,\phi) = ||Rz(t)||^2 + \int_{t_0}^{T} (||Qz(t)||^2 + |u(t)|^2)dt \qquad (8)$$

subject to

$$z(t) = T(t-t_0)\phi + \int_{t_0}^{t} T(t-s)\tilde{B}u(s)ds \ , \ z(t_0) = \varphi, \qquad (9)$$

where R is a linear bounded operator in M_2 and $Q:M([-r,0];\mathbb{R}^n) \to \mathbb{R}^p$ is such that when restricted to $C([-r,0];\mathbb{R}^n)$ it is continuous and T(t) is the solution map.

Theorem 6: The optimal solution u^0 to (8) is given by the linear feedback law of the form;

$$u^0(t) = -\tilde{B}^* P(t) z^0(t) \quad , \quad t_0 \le t \le T,$$

where $z^0(t)$ is a mild solution to an evolution equation in M_2

$$\frac{d}{dt} z(t) = (\tilde{A} - \tilde{B}\tilde{B}^* P(t))z(t) \quad , \quad z(0) = \varphi,$$

i.e. $z^0(t) = \mathcal{U}(t,t_0)\phi$, $\mathcal{U}(\cdot,\cdot)$ is the perturbation evolution operator of the semigroup $\{T(t), t \ge 0\}$ by $-\tilde{B}\tilde{B}^* P(t)$. P(t), $t_0 \le t \le T$ is the unique solution to the Riccati equation

$$\frac{d}{dt} \langle P(t)x,y \rangle = -\langle \tilde{A}x,P(t)y \rangle - \langle P(t)x,\tilde{A}y \rangle + \langle \tilde{B}^* P(t)x,\tilde{B}^* P(t)y \rangle - \langle Qx,Qy \rangle,$$

$$P(T) = R^* R \quad , \quad \text{for all } x,\ y \in \mathcal{D}(\tilde{A}). \qquad (10)$$

Moreover we have the representation of P(t),

$$P(t)\phi = \int_t^T T^*(s-t)Q^* Q\mathcal{U}(s,t)\phi ds + T^*(T-t)R^* R\mathcal{U}(T,t)\phi \text{ for all } \phi \in M_2,$$

and the interpretation of the optimality

$$\langle P(t)\phi,\phi \rangle = \inf\{J(t,T;u,\phi) \,|\, u \in L_2([t,T];\mathbb{R}^m)\} \text{ for all } \phi \in M_2.$$

Theorem 7: Consider the equation in P

$$\langle \tilde{A}x,Py \rangle + \langle Px,\tilde{A}y \rangle - \langle \tilde{B}^* Px,\tilde{B}^* Py \rangle + \langle Qx,Qy \rangle = 0$$

$$\text{for all } x,\ y \in \mathcal{D}(\tilde{A}). \qquad (11)$$

$P \in \mathcal{L}(M)$, $P^* = P$ and $P \ge 0$.

i) If $(\tilde{A},\tilde{B})$ is stabilizable, then (11) has a solution.

ii) If $(\tilde{A},Q)$ is detectable, then (11) has at most one solution, and if P^∞ is the solution, $\tilde{A}-\tilde{B}\tilde{B}^* P^\infty$ generates a stable semigroup.

iii) If $(\tilde{A},\tilde{B})$ is stabilizable and $(\tilde{A},Q)$ is detectable, then for any $R \in \mathcal{L}(M_2)$, $P_T(0) \to P^\infty$ (strongly) as $T \to \infty$, where $P_T(\cdot)$ is the unique solution to (10) starting from $P_T(T) = R^*R$ and P^∞ is the unique solution to (11). The optimal solution to infinite-time LQR problem is given by

$$u^0(t) = -\tilde{B}^* P^\infty z^0(t)$$

where $z^0(t)$ is the mild solution to

$$\frac{d}{dt} z(t) = (\tilde{A} - \tilde{B}\tilde{B}^* P^\infty) z(t) \quad , \quad z(0) = \phi .$$

REFERENCES

[1] London, W. P., and Yorke, J. A., "Recurrent Epidemics of Measles, Chickenpox, and Mumps I: Seasonal Variation in Contact Rates", American Journal of Epidemiology, Volume 98, 453-468, (1973).

[2] Boffi, V., and Scozzafava, R., "A First Order Linear Differential Difference Equation with N Delays", Journal of Mathematical Analysis and Applications, Volume 17, 577-589 (1967).

[3] Kamen, "On an Algebraic Theory of Systems Defined by Convolution Operators", Mathematical Systems Theory, Volume 9, No. 1, 57-74 (1975).

[4] Brayton, R., "Small Signal Stability Criterion for Electrical Networks Containing Lossless Transmission Lines", IBM Journal of Research and Development, Volume 12, 431-440 (1968).

[5] Brayton, R. K., and Miranker, W. L., "A Stability Theory for Nonlinear Mixed Initial-Boundary Volue Problems", Archives of Rational Mechanics and Analysis, Volume 17, 358-376 (1964).

[6] Cooke, K. L., and Krumme, D. W., "Differential-Difference Equations and Non-linear Initial-Boundary Value Problems for Linear Hyperbolic Partial Differential Equations", Journal of Mathematical Analysis and Applications, Volume 24, 372-387 (1968).

[7] Morse, A. S., "Ring Models for Delay-Differential Equations", Automatica, Volume 12, 529-531, (1976).

[8] Balakrishnan, A. V., Applied Functional Analysis, Springer-Verlag, Berlin 1976.

[9] Curtain, R. F., and Pritchard, A. J., Infinite Dimensional Linear Systems Theory, Lecture Notes in Information and Control, Springer-Verlag, 1979.

[10] Manitius, A., and Triggiani, R., "Function Space Controllability of Linear Retarded Systems: A Derivation from Abstract Operator Conditions", SIAM J. Control and Optimization, Vol. 16, pp. 599-645, 1978.

[11] Delfour, M. C., McCalla, C., and Mitter, S. K., "Stability and the Infinite-Time Quadratic Cost Problem for Linear Heriditory Differential Systems", Ibid, Vol. 13, pp. 48-88, 1975.

[12] Banks, H. T., and Kent, G., "Control of Functional Differential Equations to Target Sets in Function Space", SIAM J. Control, Vol. 10, pp. 567-593, 1972.

[13] Banks, H. T., and Jacobs, M. Q., "Attainable Sets Approach to Optimal Control of Functional Differential Equations with Function Space Constraints", J. Diff. Equations, Vol. 13, pp. 127-149, 1973.

[14] Datko, R., "Neutral Autonomous Functional Differential Equations with Quadratic Cost", SIAM J. Control, Vol. 12, pp. 70-82, 1974.

[15] Balakrishnan, A. U., "Boundary Control of Parabolic Equations: L R-Theory", Proceeding of the 5th International Summer School on Theory of Nonlinear Operators, Springer-Verlag, Berlin, pp. 12-23, 1978.

[16] Zabczyk, J., "Remark on the Algebraic Riccati Equations in Hilbert Spaces", Applied Math. and Optimization, Vol. 2, pp. 251-258, 1976.

[17] Spong, M. W., D.Sc. Thesis, Realization and Feedback Stabilization of Linear Neutral Systems, Washington University, St. Louis, Missouri 63130, 1981.

[18] Ito, K., D.Sc. Thesis, Linear Functional Differential Equations and Control and Estimation Problems, Washington University, St. Louis, Missouri 63130, 1981.

[19] Byrnes, C. I., "On the Control of Certain Infinite-Dimensional Linear Systems by Algebro-Geometric Techniques", American Journal of Mathematics, Volume 100, 1333-1381 (1978).

[20] Hale, J. K., Functional Differential Equations, Springer-Verlag, New York, (1972).

[21] Bumby, Sontag, Sussmann, and Vasconcelos, "Remarks on the Pole-Shifting Problem Over Rings", Journal of Pure and Applied Algebra, Volume 20, (1980).

[22] Spong, M. W., Byrnes, C. I., and Tarn, T. J., "A Banach Algebra Approach to the Stabilization of Neutral Delay-Differential Systems", to be submitted.

CONTROLLABILITY OF LINEAR SYSTEMS IN THE PRESENCE OF NONSEPARABLE UNCERTAINTY

B. ROSS BARMISH
Department of Electrical Engineering
University of Rochester
Rochester, New York 14627

Abstract: A dynamical system having state $x(t)\varepsilon R^n$, control $u(t)\varepsilon\Omega\subset R^m$ and uncertainty $q(t)\varepsilon Q\subset R^p$ is described by an equation of the form $\dot{x}(t) = A(t)x(t)+h(t,u(t),q(t))$. Within this framework, we consider the problem of steering an arbitrary initial state x_0 to within any given ε of a prescribed closed, convex target set X. If such a transfer of state can be guaranteed for all admissible variations of $q(\cdot)$ and all initial states x_0, then the system is said to be globally approximately (Ω,Q)-controllable to X. In this paper, we develop criteria for approximate global (Ω,Q)-controllability to X without assuming that $h(\cdot)$ is separable; that is, in contrast to previous work, we do not require $h(\cdot)$ to be decomposable as a sum $f(t,u(t))+g(t,q(t))$.

I. INTRODUCTION

This paper is primarily concerned with problems of controllability for a class of uncertain dynamical systems described by

$$\dot{x}(t) = A(t)x(t)+h(t,u(t),q(t));\ t\varepsilon[0,\infty) \tag{S}$$

where $x(t)\varepsilon R^n$ is the state, $u(t)\varepsilon\Omega\subset R^m$ is the control, $q(t)\varepsilon Q\subset R^p$ is the uncertain parameter vector (or disturbance), $A(t)\varepsilon R^{n\times n}$ is the state matrix and $h:[0,\infty)\times R^m\times R^p\to R^n$ describes the effects of the control and uncertainty on the system. In addition, a target set $X\subset R^n$ will be taken as given. For simplicity, X is assumed to be stationary. We note, however, that the development to follow can be easily modified to accommodate a moving target $X(t)$. The regularity assumptions on $A(\cdot)$, $h(\cdot)$, Ω, Q and X will be stated in the sequel.

For uncertain dynamical systems of the sort described above, we shall adopt a notion of approximate (Ω,Q)-controllability to X; e.g., see [1]. The objective will be to decide whether or not it is possible, by choice of admissible control $u(\cdot)$, to steer the state of the system to the target in a so-called guaranteed sense. More precisely, knowing only that the uncertain parameter vector takes its values $q(t)$ in Q, one wishes to provide a guarantee that the state can be steered to within an arbitrarily small distance from the given target X in a finite amount of time. It is important to emphasize that knowledge of the time-varying realization of $q(\cdot)$ is not assumed. Only the bounding set Q is available to the controller. Moreover, the possibly rapid time variations in $q(\cdot)$ precludes parameter estimation as in the pursuit game scenario; e.g., see [3].

In order to relate this work to existing literature, one can look at the special case obtained when one assumes separability of the function $h(\cdot)$ which enters into the dynamics of the system (S). More precisely, when $h(\cdot)$ is decomposable as

$$h(t,u(t),q(t)) = f(t,u(t))+g(t,q(t)) \tag{1.1}$$

where $f(\cdot)$ and $g(\cdot)$ are appropriate functions on $[0,\infty)\times R^m$ and

$[0,\infty) \times R^p$ respectively, then we are dealing with the case of separable uncertainty; e.g., see [1], [4] and [5]. In the fuller version of this paper [14], it will be shown how one can obtain the results of these papers as special cases of theorems given here. Furthermore, for the case of separable uncertainty, it is seen in [14] that the notion of approximate (Ω,Q)-controllability is in fact equivalent to the notion of (exact) (Ω,Q)-controllability used in [1], [4] and [5]. Currently, this author is attempting to develop more general (and weaker) conditions on $A(\cdot)$, $h(\cdot)$, Ω, X and Q under which approximate and exact (Ω,Q)-controllability are equivalent.

If one further specializes by taking the bounding set Q to be a singleton $\{q_0\}$, then (S) is no longer uncertain and the present theory degenerates into that of [2]. In this way, the work described here is easily related to the rather large body of literature on controllability theory for deterministic systems; e.g., see the bibliography of [2]. To make this link with the existing literature, we shall show in [14] how the results given here degenerate into those of [1] and [2] under the imposition of additional hypotheses. The reader is referred to [2] to see how one can further specialize to obtain existing controllability criteria; e.g., see [6]-[11].

II. REGULARITY ASSUMPTIONS

In order to assure existence and uniqueness of state equation solutions and well definedness of certain mathematical operations, we impose a set of standard regularity assumptions. In the sequel, the following assumptions will be used wherever required.

Assumptions:

A1. The control set Ω is compact.
A2. The bounding set Q is compact.
A3. The target set X is closed and convex.
A4. The entries of the matrix function $A(\cdot)$ are Lebesgue measurable and integrable over finite sub-intervals of $[0,\infty)$.
A5. For each fixed pair $u(\cdot)\varepsilon M(\Omega)$ and $q(\cdot)\varepsilon M(Q)$, the entries of the vector function $h(\cdot,u(\cdot),q(\cdot))$ are Lebesgue measurable and integrable over finite sub-intervals of $[0,\infty)$.
A6. For each fixed $t\varepsilon[0,\infty)$, $h(t,\omega,q)$ is continuous with respect to $(\omega,q)\varepsilon\Omega\times Q$.

III. NOTATION AND DEFINITIONS

Notation: The state transition matrix of the unforced system $\dot{x}(t) = A(t)x(t)$ will be denoted by $\phi(t,\tau)$ and the adjoint system will have dynamics described by

$$\dot{z}(t) = -A'(t)z(t); \quad t\varepsilon[0,\infty). \tag{S'}$$

To denote the fact that an admissible control $u(\cdot)$ is Lebesgue measurable and takes its values $u(t)$ in Ω, we shall write "$u(\cdot)\varepsilon M(\cdot)$." Similarly, the notation "$q(\cdot)\varepsilon M(Q)$" will be used to describe admissible uncertainties. Hence, given a time $T\varepsilon[0,\infty)$, an initial condition $x(0) \triangleq x_0\varepsilon R^n$, an admissible control $u(\cdot)\varepsilon M(\Omega)$ and an admissible uncertainty $q(\cdot)\varepsilon M(Q)$, the resulting state at time T will be written as $x(T,x_0,u(\cdot),q(\cdot))$. A sequence

$$Y \triangleq \{y_i\}_{i=1}^{\infty}$$

in R^k is said to be of finite extent if there is a positive integer N such that $y_i = 0$ for all $i \geq N$. The space of all such sequences will be denoted by $\mathcal{R}_k^\infty$ and, for $k = 1$ we can define a simplex of countable dimension

$$S^\infty \triangleq \{S \triangleq \{s_i\}_{i=1}^\infty \varepsilon \mathcal{R}_1^\infty : s_i \geq 0 \text{ for all } i;\ \sum_i s_i = 1\}.$$

Also, for any positive integer k, one can also define the k-dimensional simplex

$$S^k \triangleq \{S \triangleq \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_k \end{bmatrix} \varepsilon R^k : s_i \geq 0 \text{ for all } i;\ \sum_{i=1}^k s_i = 1\}.$$

Finally, if V is a subset of R^k and $\tilde{v} \varepsilon R^k$, then the distance from $\tilde{v}$ to V is given by

$$d(\tilde{v},V) \triangleq \inf_{v \varepsilon V} \|v-\tilde{v}\|$$

where $\|\cdot\|$ is the chosen norm on R^k. The results to follow will not depend on which norm is chosen for the state space R^n.

Definition 3.1: Given $x_0 \varepsilon R^n$ and $T \varepsilon [0,\infty)$, (S) is said to be approximately (Ω,Q)-controllable to X from x_0 at time T if the following condition holds: Given any $\varepsilon > 0$, there is a control $u_\varepsilon(\cdot) \varepsilon M(\Omega)$ such that

$$d(x(T,x_0,u_\varepsilon(\cdot),q(\cdot)),X) \leq \varepsilon \ \text{ for all } q(\cdot) \varepsilon M(Q).$$

If the condition above can be made to hold with $\varepsilon = 0$, then (S) is said to be (exactly) (Ω,Q)-controllable to X at time T; e.g., see [1], [4] and [5]. Letting $u_0(\cdot)$ replace $u_\varepsilon(\cdot)$ for the case when $\varepsilon = 0$, the requirement above is tantamount to

$$x(T,x_0,u_0(\cdot),q(\cdot)) \varepsilon X \text{ for all } q(\cdot) \varepsilon M(Q).$$

Finally, we say that (S) is approximately (Ω,Q)-controllable to X from x_0 if there exists some $T \varepsilon [0,\infty)$ such that (S) is approximately (Ω,Q)-controllable to X from x_0 at time T.

Definition 3.2: (S) is said to be globally approximately (Ω,Q)-controllable to X if (S) is approximately (Ω,Q)-controllable to X from every $x_0 \varepsilon R^n$. Similarly, if for every $x_0 \varepsilon R^n$, the above condition can be made to hold with $\varepsilon = 0$, then (S) is said to be (exactly) globally (Ω,Q)-controllable to X.

Remarks: In effect, the preceding definitions require guaranteed performance of the system. When the control $u_\varepsilon(\cdot)$ in Definition 3.1 is applied, it is "guaranteed" that the state will eventually get to within ε of the target. Such a guarantee is given independently of the uncertainty $q(\cdot) \varepsilon M(Q)$. It is interesting to note that for systems without uncertainty, one need not distinguish between approximate and exact controllability; e.g. see [2] and [6]-[11]. That is, for this case, it can be readily shown that these two notions of controllability are equivalent. The proof of this equivalence is accomplished using the closedness property of the so-called attainable set; e.g., see [12], [13]. For uncertain dynamical systems, however, it is presently unknown whether Assumptions A1-A6 are sufficient in order to establish equivalence between approximate and exact (Ω,Q)-controllability.

IV. PRELIMINARY RESULTS

In order to prepare for the proof of our main theorem on global approximate (Ω,Q)-controllability, we shall develop some preliminary results. In light of the page limitation, the proof of following "geometrically obvious" lemma is relegated to [14], a fuller version of this paper.

Lemma 4.1: *Let* $x_0 \varepsilon R^n$ *be given. Then* (S) *is approximately* (Ω,Q)-*controllable to* X *from* x_0 *if and only if the following condition is satisfied:* (S) *is approximately* (Ω,Q)-*controllable to every polyhedral superset* $X_+ \supseteq X$ *from* x_0.

In view of Lemma 4.1, it will be possible to work with linear inequalities when performing computations involving the target. This will enable us to use the theorems of [15] and [16] dealing with "robustness" of perturbed integral equations. To simplify the exposition, we interpret the results of [15] and [16] using the terminology and notation of this paper.

Theorem 4.2 (see [15] or [16] for proof): *Given* $T \varepsilon [0,\infty)$, *consider the dynamical system* (S) *for the special case obtained when* $A(t) \equiv 0$, $x_0 = 0$ *and* X *is a polyhedral set described by the linear inequalities*

$$a_i'x \le b_i; \quad i = 1,2,\ldots,p \tag{4.1}$$

where $a_i \varepsilon R^n$ *and* $b_i \varepsilon R$ *for* $i = 1,2,\ldots,p$. *Then* (S) *is approximately* (Ω,Q)-*controllable to* X *from* x_0 *at time* T *if and only if the following condition holds for all* $S \varepsilon S^p$:

$$h(S) + \int_0^T H(S,\tau)d\tau \ge 0 \tag{4.2}$$

where $h\colon S^p \to R$ *and* $H\colon S^p \times [0,\infty) \to R$ *are given by*

$$h(S) \triangleq \sum_{i=1}^{p} s_i b_i; \tag{4.3}$$

$$H(S,\tau) \triangleq \max_{\omega\varepsilon\Omega} \sum_{i=1}^{p} s_i \min_{q\varepsilon Q} - a_i'h(\tau,\omega,q).$$

Although Theorem 4.2 addresses the special case obtained when $A(t) \equiv 0$, $x_0 = 0$ and X is polyhedral, one can easily remove these restrictions by taking note of the following:

(i) Incorporation of a non-zero initial state is tantamount to translation of the target by the fixed vector $\phi(T,0)x_0$.

(ii) One can handle non-zero $A(t)$ by simply replacing $h(\tau,\omega,q)$ by $\phi(T,\tau)h(\tau,\omega,q)$ in the expression for $H(S,\tau)$.

(iii) A compact convex set can be approximated arbitrarily closely by a polyhedral convex set. Hence, to accommodate an arbitrary closed convex target X, we first consider the intersection of the target X with so-called *attainable set*

$$\pi \triangleq \{\phi(T,0)x_0 + \int_0^T \phi(T,\tau)h(\tau,u(\tau),q(\tau))d\tau :\ u(\cdot)\epsilon M(\Omega),\ q(\cdot)\epsilon M(Q)\}.$$

Noting that π is compact and convex (see [12] or [13]), it follows that $\hat{X} \triangleq \pi \cap X$ is also compact and convex. Subsequently, one can apply the preceding theorem to a sequence of polyhedral approximations which are approaching $\hat{X}$ (which can be used in lieu of X).

The theorem to follow, a generalization of Lemma 3.3 of [1] and Theorem 5.1' of [2], is proven using observations (i)-(iii) above. A fully detailed proof is given in [14]. To state our result, we define a mapping $J: R^n \times [0,\infty) \times \mathcal{R}_n^\infty \times S^\infty \to R \cup \{+\infty\}$ as follows: Given $x_0 \epsilon R^n$, $T\epsilon[0,\infty)$, $Z(0) \triangleq \{z_i(0)\}_i \epsilon \mathcal{R}_n^\infty$ and $S \triangleq \{s_i\}_i \epsilon S^\infty$, let

$$J(x_0,T,Z(0),S) \triangleq \sum_i s_i x_0' z_i(0) + \int_0^T \max_{\omega\epsilon\Omega} \sum_i s_i \min_{q\epsilon Q} h'(\tau,\omega,q) z_i(\tau)d\tau - \sum_i s_i \inf_{x\epsilon X} x' z_i(T) \tag{4.4}$$

where $z_i(\cdot)$ is the solution of the adjoint system (S') resulting from initial condition $z_i(0)$.

Theorem 4.3: *Let* $x_0 \epsilon R^n$ *and* $T\epsilon[0,\infty)$ *be given. Then* (S) *is approximately* (Ω,Q)*-controllable to* X *from* x_0 *at time* T *if and only if*

$$J(x_0,T,Z(0),S) \geq 0 \tag{4.5}$$

for all pairs $(Z(0),S)\epsilon\mathcal{R}_n^\infty \times S^\infty$.

V. MAIN RESULTS ON GLOBAL APPROXIMATE (Ω,Q)-CONTROLLABILITY

In [1], [2] and [5], the so-called separable version of this problem was considered. The results on global controllability were described in terms of a pair of functions $V: R^n \times [0,\infty) \to R \cup \{+\infty\}$ and $W: [0,\infty) \to R \cup \{+\infty\}$. Our objective here is to describe more general "V and W functions" which will enable one to analyze the case involving nonseparable uncertainty. To this end, we define three functions $U: \mathcal{R}_n \times S^\infty \times [0,\infty) \to R \cup \{+\infty\}$, $V: \mathcal{R}_n^\infty \times [0,\infty) \to R \cup \{+\infty\}$ and $W: [0,\infty) \to R \cup \{+\infty\}$ given by

$$U(Z(0),S,t) \triangleq \int_0^t \max_{\omega\epsilon\Omega} \sum_i s_i \min_{q\epsilon Q} h'(\tau,\omega,q) z_i(\tau)d\tau - \sum_i s_i \inf_{x\epsilon X} x' z_i(t); \tag{5.1}$$

$$V(Z(0),t) \triangleq \sup_{S\epsilon S^\infty} U(Z(0),S,t); \tag{5.2}$$

$$W(t) \triangleq \inf_{S\epsilon S^\infty,\ \|Z(0)\|=1} U(Z(0),S,t) \tag{5.3}$$

where $z_i(\cdot)$ is the solution of the adjoint system (S') resulting from initial condition $z_i(0)$ and if $Z\epsilon\mathcal{R}_n^\infty$, we take

$$\|Z\| \triangleq \max_i \|z_i\|$$

with $\|z_i\|$ taken to be any norm on R^n. We are now prepared to state our main result. The theorem below is a generalization of Theorem 5.2 of [1] and Theorem 4.1 of [2].

Theorem 5.1: The following condition is necessary for global approximate (Ω,Q)*-controllability to* X:

$$\sup_{t\geq 0} V(Z(0),t) = +\infty \tag{5.4}$$

for all nonzero initial conditions $Z(0)\varepsilon\mathcal{R}_n^\infty$. *The following condition is sufficient for global approximate* (Ω,Q)*-controllability to* X:

$$\sup_{t\geq 0} W(t) = +\infty. \tag{5.5}$$

Proof: (Necessity) Proceeding by contradiction, suppose that (S) is globally approximately (Ω,Q)-controllable to X but

$$\sup_{t\geq 0} V(\tilde{Z}(0),t) = \beta < +\infty$$

for some non-zero $\tilde{Z}(0)\varepsilon\mathcal{R}_n^\infty$ and some finite positive constant β. Writing $\tilde{Z}(0) \triangleq \{\tilde{z}_i(0)\}_{i=1}^k$, it follows that $\tilde{z}_\ell(0) \neq 0$ for some $\ell \leq k$. Next, we define the initial condition

$$\tilde{x}_0 \triangleq \frac{-2\beta\tilde{z}_\ell(0)}{\|\tilde{z}_\ell(0)\|^2}$$

and the vector $\tilde{S}\varepsilon S^k$ by

$$\tilde{s}_k \triangleq \begin{cases} 0 & \text{if } k\neq\ell; \\ 1 & \text{if } k=\ell. \end{cases}$$

Then, given any $T\varepsilon[0,\infty)$ one can substitute $\tilde{x}(0)$ and $\tilde{Z}(0)$ into (4.4) and obtain

$$J(\tilde{x}_0,T,\tilde{Z}(0),\tilde{S}) = -2\beta + \int_0^T \max_{\omega\varepsilon\Omega}\min_{q\varepsilon Q} h'(\tau,\omega,q)\tilde{z}_\ell(\tau)d\tau - \inf_{x\varepsilon X} x'\tilde{z}_\ell(T)$$

$$\leq -2\beta + V(\tilde{Z}(0),T) \leq -\beta < 0.$$

By Theorem 4.3, this implies that (S) cannot be globally approximately (Ω,Q)-controllable to X (a contradiction).

(Sufficiency) Proceeding again by contradiction, we assume that

$$\sup_{t\geq 0} W(t) = +\infty$$

but (S) is not globally approximately (Ω,Q)-controllable to X. Hence, in light of Theorem 4.3, there exists an initial condition $\tilde{x}_0$ having the following property: Given any $t\varepsilon[0,\infty)$, there is a pair $(Z^t(0),S^t)\varepsilon\mathcal{R}_n^\infty\times S^\infty$ such that $\|Z^t(0)\| = 1$ and $J(\tilde{x}_0,t,Z^t(0),S^t) < 0$. Noticing that

$$J(\tilde{x}_0,t,Z^t(0),S) = \sum_i s_i^t \tilde{x}_0' z_i^t(0) + U(Z^t(0),S^t,t).$$

we obtain the following bound for each fixed $t \varepsilon [0,\infty)$:

$$U(Z^t(0),S^t,t) \leq -\sum_i s_i^t \tilde{x}_0' z_i^t(0) \leq \sum_i s_i^t |\tilde{x}_0' z_i^t(0)|$$

$$\leq \|\tilde{x}_0\| \max_i \|z_i^t(0)\| = \|\tilde{x}_0\|. \tag{5.6}$$

Hence, using (5.6), we generate an upper bound for $W(t)$. That is, given any $t \varepsilon [0,\infty)$, we have

$$\|\tilde{x}_0\| \geq U(Z^t(0),S^t,t) \geq W(t).$$

This clearly contradicts (5.5). □

VI. REFERENCES

[1] B.R. Barmish, "Point and Global Controllability of Uncertain Dynamical Systems," IEEE Transactions on Automatic Control, in press.

[2] ______ and W.E. Schmitendorf, "New Results on Controllability of Systems of the Form $\dot{x}(t) = A(t)x(t) + f(t,u(t))$," IEEE Transactions on Automatic Control, AC-25, pp. 540-547, 1980.

[3] O. Hajek, Pursuit Games, Academic Press, New York, 1975.

[4] B.R. Barmish, Y.H. Lin and J. Sankaran, "Reachability of Targets Subject to Disturbances Bounded by Convex Sets," International Journal of Control, Vol. 28, pp. 771-783, 1978.

[5] ______ and W.E. Schmitendorf, "The Associated Disturbance-Free System," Journal of Optimization Theory and Applications, to appear.

[6] W.E. Schmitendorf and B.R. Barmish, "Null Controllability of Linear Systems with Constrained Controls," SIAM Journal of Control and Optimization, Vol. 18, pp. 327-345, 1980.

[7] J.P. Dauer, "On Controllability of Systems of the Form $\dot{x} = A(t)x + g(t,u)$," Journal of Optimization Theory and Applications, Vol. 11, pp. 132-138, 1973.

[8] S. Saperstone and J. Yorke, "Controllability of Linear Oscillatory Systems Using Positive Controls," SIAM Journal of Control, Vol. 9, pp. 253-262, 1971.

[9] R. Conti, Teoria del Controllo e del Controllo Ottimo, UTET, Torino, Italy, 1974.

[10] L. Pandolfi, "Linear Control Systems: Controllability with Constrained Controls," Journal of Optimization Theory and Applications, Vol. 19, pp. 577-585, 1976.

[11] R. Brammer, "Controllability of Linear Autonomous Systems with Positive Controllers," SIAM Journal of Control, Vol. 10, pp. 339-353, 1972.

[12] L.W. Neustadt, "The Existence of Optimal Controls in the Absence of Convexity," Journal of Mathematical Analysis and Applications, Vol. 8, pp. 110-117, 1963.

[13] C. Olech, "Extremal Solutions of a Control System," Journal of Differential Equations, Vol. 2, No. 1, pp. 74-101, 1966.

[14] B.R. Barmish, "Controllability Criteria for the Uncertain Dynamical System $\dot{x}(t) = A(t)x(t) + h(t,u(t),q(t))$," in preparation.

[15] _____, L.E. Blume and S.D. Chikte, "Robustness of Systems with Uncertainties in the Input," Journal of Mathematical Analysis and Applications, in press.

[16] _____, "On a Class of Nonlinear Robustness Problems," Proceedings of the 16th Allerton Conference on Communication, Control and Computing, Urbana-Champaign, 1979.

[17] S.D. Chikte and B.R. Barmish, "Some Fundamental Problems Associated with Guaranteed Controllability of Uncertain Systems," Proceedings of the Joint Automatic Control Conference, San Francisco, 1980.

COMPUTATIONAL ASPECTS OF DISCRETE TIME ADAPTIVE CONTROL[1]

R. CRISTI and R. V. MONOPOLI
Electrical and Computer Engineering Department, University of Massachusetts, Amherst, Massachusetts 01003

ABSTRACT

Stable Model Reference Adaptive Control schemes for discrete time systems can be designed which take into account computational time and limited range of computing machines. An algorithm is developed using Lyapunov second method and properties of strictly positive real functions.

INTRODUCTION

An increasing part of the current literature on control systems is dedicated to discrete time adaptive control schemes. This emphasis can be attributed to advances in digital microelectronic components which are allowing more and more versatile applications at decreasing prices. However, many of the discrete control schemes reported, as for example in [1], [4], and [5], are difficult to implement for the following reasons:

-- they do not leave any time for computations, i.e., the action of the controller at any instant k requires samples of the system at the same instant k;

-- the adaptive algorithm is a function of quantities which tend to zero as time goes to infinity, causing problems of under-flow in limited word-length machines such as microprocessors. In order to compensate for inconveniences of this kind, time variable adaptive gains have been introduced in [4], [5], but these either suffer from too rigid constraints, as in [4], or from a too limited range of variation, as in [5].

The aim of this paper is to introduce a discrete time adaptive control algorithm for single input-single output systems which is more suitable for practical implementation, and which does not have drawbacks discussed above.

The paper work is divided into four parts: problem statement, main mathematical results, adaptive control implementation, and updating algorithm.

NOTATION

The following notation will be used:

-- lower case for scalars, real in general: a;

-- vectors: $\underline{a} = [a_1, a_2, \ldots, a_n]^T$;

-- matrices: upper case, e.g., A; I is the identity matrix;

-- time delay operator: z^{-1};

-- n-degree polynomials in z^{-1}: $Z_a(n) = a_0 + \sum_{i=1}^{n} a_i z^{-i}$;

-- rational functions in z^{-1}: $H(z^{-1}) = \dfrac{Z_a(n)}{Z_b(m)}$

-- conditions on polynomials and rational functions in z^{-1} will be given with respect to the z-plane

---

[1] This research was supported in part by the Air Force Office of Scientific Research under Grant Number AFOSR-80-0155.

I. PROBLEM STATEMENT

A discrete time dynamic system (plant) can be described by the linear, time invariant, non autonomous difference equation

$$x(k+n) + a_1 x(k+n-1) + \dots + a_n x(k) = b_0\, u(k+m) + \dots + b_m\, u(k) \qquad (1.1)$$

where $\{u(k)\}$, $\{x(k)\}$, $k = 1,2, \dots$ are the plant input and output, respectively.

The following assumptions are made on the plant parameters:

- (i) the values of a_i, $i=1,n$ and b_i, $i=0,m$ are unknown;
- (ii) $m \leq n-1$ are known;
- (iii) the plant is minimum phase, i.e., the polynomial $b_0 + b_1 z^{-1} + \dots + b_m z^{-m}$ is analytical outside the unit circle in the z-plane;
- (iv) the sign of b_0 is known, as are bounds b_{0M} and b_{0m}, where $b_{0M} \geq b_0 \geq b_{0m}$. Without loss of generality, $b_{0m} > 0$ will be assumed.

The design objective is to have the plant output follow the output of a model reference defined by the equation

$$x^M(k+n) + a_{d_1} x^M(k+n-1) + \dots + a_{d_n} x^M(k) = K_0 r(k) \qquad (1.2)$$

Introducing the delay operator z^{-1} and the notation described in Section 0, (1.1) and (1.2) can be written as

$$Z_p(n) x(k) = z^{-d}\, Z_u(m)\, u(k) \qquad (1.1\text{-A})$$

$$Z_M(n) x^M(k) = z^{-n} K_0 r(k) \qquad (1.2\text{-A})$$

where $d \triangleq n - m$.

The model will be supposed asymptotically stable, i.e., $Z_M(n)$ has roots inside the unit circle in the z-plane.

II. MAIN THEOREM

The arguments given in the following sections will rely mainly on the following theorem:

Theorem 1. Given the non-linear difference equation

$$H(z^{-1})\eta(k) = \underline{\delta}^T(k)\underline{\phi}(k) + q(k) \qquad (2.1)$$

where $\{\underline{\phi}(k)\}_{k=1}^{\infty}$ is any sequence of vectors and $\underline{\delta}(k)$ and $q(k)$ are sequences which satisfy

$$\underline{\delta}(k+1) = \underline{\delta}(k) - F(k)\underline{\phi}(k)\eta(k) \qquad (2.2)$$

$$q(k) = -\alpha(k)\, \| \underline{\phi}(k) \|^2\, \eta(k) \qquad (2.3)$$

where $F(k)$ and $\alpha(k)$ bounded and are such that

$[F^{-1}(k) - F^{-1}(k+1)]$ is semipositive definite for all $k \geq 1$ and $F(0)$ is positive definite; (2.4)

$[2\alpha(k)I - F(k)]$ is positive definite for all $k \geq 0$ and (2.5)
$H(z^{-1})$ is Strictly Positive Real (SPR),

then it follows that

$$\lim_{k\to\infty} \eta(k) = 0 \qquad \lim_{k\to\infty} \eta(k)\underline{\phi}(k) = \underline{0} \tag{2.6}$$

$$\lim_{k\to\infty} \underline{\delta}(k-1) - \underline{\delta}(k) = \underline{0} \tag{2.7}$$

Proof: Let us choose as a potential Lyapunov function for the system (2.1)

$$V(k) = \underline{\delta}(k)\ F^{-1}(k)\ \underline{\delta}(k) + \underline{x}^T(k)\ P\underline{x}(k) \tag{2.8}$$

where $\underline{x}(k)$ is the state sequence of any minimal realization (A, B, C, J) of $H(z^{-1})$, and $P = P^T$ is a positive definite matrix which will be determined later.

From (2.2) it follows that

$$\underline{\delta}^T(k+1)F^{-1}(k)\underline{\delta}(k+1) = \underline{\delta}^T(k)F^{-1}(k)\underline{\delta}(k) - 2\underline{\delta}^T(k)\underline{\phi}(k)\eta(k) + + \underline{v}^T(k)F(k)\underline{\phi}(k)\eta^2(k) \tag{2.9}$$

Substituting (2.1) in (2.9) for $\underline{\delta}^T(k)\underline{\phi}(k)$, we get

$$\underline{\delta}^T(k+1)F^{-1}(k)\underline{\delta}(k+1) - \underline{\delta}^T(k)F^{-1}(k)\underline{\delta}(k) = -2[H(z^{-1})\eta(k)]\eta(k) + + 2q(k)\eta(k) + \underline{v}^T(k)F(k)\underline{\phi}(k)\eta^2(k) \tag{2.10}$$

Since $H(z^{-1})$ is SPR, it is shown in Appendix A that there exist matrices P, positive definite, and L and W such that

$$2[H(z^{-1})\eta(k)]\eta(k) = \underline{x}^T(k+1)P\underline{x}(k+1) - \underline{x}^T(k)P\underline{x}(k) + + \|L^T\underline{x}(k) + W\eta(k)\|^2 \tag{2.11}$$

Substituting (2.3) and (2.11) into (2.10) yields

$$\underline{\delta}^T(k+1)F^{-1}(k)\underline{\delta}(k+1) - \underline{\delta}^T(k)F^{-1}(k)\underline{\delta}(k) + \underline{x}^T(k+1)P\underline{x}(k+1) - - \underline{x}^T(k)P\underline{x}(k) = -\underline{v}^T(k)[2\alpha(k)I - F(k)]\underline{\phi}(k)\eta^2(k) - \|L^T\underline{x}(k) + W\eta(k)\|^2 \tag{2.12}$$

Since condition (2.4) yields

$$\underline{\delta}^T(k+1)F^{-1}(k+1)\underline{\delta}(k+1) \leq \underline{\delta}^T(k+1)F^{-1}(k)\underline{\delta}(k+1),$$

then substitution of (2.8) into (2.12) yields

$$V(k+1) - V(k) \leq -\underline{v}^T(k)[2\alpha(k)I - F(k)]\underline{\phi}(k)\eta^2(k) - - \|L^T\underline{x}(k) + W\eta(k)\|^2 \leq 0. \tag{2.13}$$

It is seen from (2.13) and condition (2.5) and V(k) is a nonnegative and nonincreasing function of k. Therefore $\lim_{k\to\infty} V(k)$ exists and $\lim_{k\to\infty} V(k) =$

$V < \infty$, given that $V(0)$ is bounded.

Moreover, since (2.13) implies that, for every k_0,

$$V - V(k_0) = \sum_{k=k_0}^{\infty} V(k+1) - V(k) \le - \sum_{k=k_0}^{\infty} \underline{v}^T(k)[2\alpha(k)I - F(k)]\underline{\phi}(k)\eta^2(k)$$

then

$$\lim_{k\to\infty} \underline{\phi}(k)\eta(k) = \underline{0}. \tag{2.14}$$

Furthermore, since $F(k)$ is bounded by definition, (2.14), (2.2) and (2.8) imply that

$$\lim_{k\to\infty} (\underline{\delta}(k) - \underline{\delta}(k+1)) = \underline{0} \tag{2.15}$$

Finally, (2.8) and (2.13) imply that $\lim_{k\to\infty} \eta(k) = \eta(\infty) < \infty$. Let us suppose $\eta(\infty) \neq 0$: then (2.14) yields

$$\lim_{k\to\infty} \underline{\phi}(k) = 0.$$

This, together with (2.1), (2.3), and $H(z^{-1})$ being analytic outside the unit circle, yields $\mu(\infty) = 0$, and then

$$\lim_{k\to\infty} \eta(k) = 0. \tag{2.17}$$

Q.E.D.

III. ADAPTIVE CONTROL IMPLEMENTATION

A. The Error Equation

The objective of the controller is to determine an input sequence $u(k)$ to the plant such that the error sequence $e(k) \triangleq x^M(k) - x(k)$ goes asymptotically to zero.

Let $Z_w(n-1) \triangleq c_0 + \sum_{i=1}^{n-1} c_i z^{-i}$ be any $(n-1)$ degree polynomial in the delay operator, analytic outside the unit circle, such that

$$W(z^{-1}) \triangleq \frac{Z_w(n-1)}{Z_M(n)} \tag{3.2}$$

is SPR. Then combining (1.1) and (1.2) yields

$$Z_M(n)e(k) = Z_w(n-1)\Big[K_0 \frac{z^{-n} r(k)}{Z_w(n-1)} - z^{-d} \frac{Z_u(m)}{Z_w(n-1)} u(k) + z^{-1} \frac{Z_\Delta(n-1)}{Z_w(n-1)} x(k)\Big], \tag{3.3}$$

where $Z_\Delta(n-1) \triangleq \sum_{i=1}^{n} \Delta a_i z^{-(i-1)}$, $\Delta a_i \triangleq a_i - a_{d_i}$ and $d \triangleq n-m > 0$. From [1], (Lemma 1.2), and (1.1-A), (3.3) can be written as

$$Z_M(n)e(k) = Z_w(n-1)\Big[-\frac{b_0}{c_0}\,u^f(k) + \sum_{i=1}^{n}\beta_u^i\phi_u(k-i) + \tag{3.5}$$

$$+ \sum_{i=1}^{n}\beta_x^i\phi_x(k-i) + K_0\phi_r(k)\Big]$$

where the following definitions pertain:

$$u^f(k) = \frac{z^{-d}}{Z_f(d-1)}\,u(k);$$

$Z_f(d-1) = 1 + \sum_{i=1}^{d-1} F_i z^{-i}$ is any asymptotically stable polynomial;

$$\phi_u(k) = \frac{z^{-d}}{Z_f(d-1)Z_w(n-1)}\,u(k) = \frac{u^f(k)}{Z_w(n-1)}\,; \tag{3.6}$$

$$\phi_x(k) = \frac{z^{-d}}{Z_f(d-1)Z_w(n-1)}\,x(k)$$

$$\phi_r(k) = \frac{z^{-n}}{Z_w(n-1)}\,r(k)$$

and β_u^i, β_x^i are (unknown) constants).

Before developing the complete solution for adaptive control purposes, it is useful to interpret the results obtained so far, assuming the plant parameters are known. In this case the constants β_x^i and β_u^i can be computed from the parameters of the plant and the model. From (3.5) it is possible to see that the input sequence to the plant

$$u(k) = -\frac{b_0}{c_0}\Big[\sum_{i=1}^{n}\beta_u^i\, Z_f(d-1)\phi_u(k+d-i) + \sum_{i=1}^{n}\beta_x^i\, Z_f(d-1)\phi_x(k+d-i) +$$

$$+ K_0 Z_f(d-1)\phi_r(k+d)\Big]$$

yields

$$Z_M(n)\,e(k) = 0;$$

then, since the model is stable, $\lim_{k\to\infty} e(k) = 0$, and $e(k) = 0$ if $e(0) = 0$.

B. The Control Input

In order to determine the control input to the plant $u(k)$, let us write $u^f(k)$ in the form

$$u^f(k) = \sum_{i=1}^{n}K_u^i(k)\phi_u(k-i) + \sum_{i=1}^{n}K_x^i(k)\phi_x(k-i) + K_r(k)\phi_r(k) - w^1(k) \tag{3.7}$$

where $K_j^i(k)$ are variable gains and $w^1(k)$ is a sequence to be determined. Substituting (3.7) into (3.5) it follows that

$$Z_M(n)e(k) = Z_w(n-1)\Big[\sum_{i=1}^{n}\delta_u^i(k)\phi_u(k-i) + \sum_{i=1}^{n}\delta_x^i(k)\phi_x(k-i) + \tag{3.8}$$

$$+ \delta_r(k)\phi_r(k) - \frac{b_0}{c_0}\,w^1(k)\Big]$$

where

$$\delta_j^i(k) \overset{\Delta}{=} -\frac{b_0}{c_0} K_j^i(k) + \beta_j^i, \quad j = u, x, r. \tag{3.8'}$$

Introducing the "augmenting network"

$$Z_M(n)y(k) = Z_w(n-1)[K_w(k)w^1(k) + q(k)] \tag{3.9}$$

and the "augmented error"

$$\eta(k) \overset{\Delta}{=} e(k) + y(k) \tag{3.10}$$

(3.8) becomes

$$Z_m(n)\eta(k) = Z_w(n-1)[\underline{\delta}^T(k)\underline{\phi}(k) + q(k)] \tag{3.13}$$

with appropriate vector definitions.

IV. UPDATING ALGORITHM

Applying the results of Theorem 1 to equation (3.13), and considering that $Z_w(n-1)/Z_M(n)$ is SPR, we can prove eq. (3.8') and (2.2), together with

$$u(k) = \sum_{i=1}^{n} K_u^i(k)Z_f(d-1)z^d\phi_u(k-i) + \sum_{i=1}^{n} K_x^i(k)Z_f(d-1)z^d\phi_x(k-i) + \tag{4.3}$$

$$+ K_r^i(k)Z_f(d-1)z^d\phi_r(k)$$

yields $e(k) \to 0$.

The adaptive gains $K_j^i(k)$ can be evaluated recursively from (2.2) as follows:

$$K_j^i(k+1) = K_j^i(k) + \frac{c_0}{\lambda_j^i(k)b_0}\phi_j(k-i)\eta(k); \quad \begin{matrix} i = 0, n \\ j = u, x, r, w \end{matrix} \tag{4.1'}$$

$$q(k) = -\alpha(k)\|\underline{\phi}(k)\|^2\eta(k);$$

with constraints

$$0 < \lambda_j^i(k+1) \leq \lambda_j^i(k) < M < \infty$$

$$\alpha(k) \geq \frac{1}{2\lambda_j^i(k)} > 0; \quad \text{for any } k.$$

Observation 1. The adaptive gains at time k do not depend on the samples of the system at the same instant k but only on measurements taken at k-1. This fact allows one complete sampling period for computation.

Observation 2. Eq. (2.7) implies that $\phi(k-i)\eta(k)$ becomes smaller as k goes to infinity, slowing down the adaptation process. Faster convergence can be achieved by decreasing the gains $\lambda_j^i(k)$.

In order to show that the error goes to zero, (3.6), (3.7), and (4.3) yield

$$Z_f(d-1)w^1(k) = Z_f(d-1)\underline{K}^T(k)\underline{\phi}(k) - \underline{K}^T(k-d)Z_f(d-1)\underline{\phi}(k), \qquad (4.4)$$

which implies $w^1(k) \to 0$, as $\underline{K}(k) \to \underline{K}(\infty)$ [1].

Furthermore, if $\underline{\phi}(k)$ is bounded [3], then from (2.2) and (2.3), it follows that

$$\lim_{k\to\infty} q(k) = 0;$$

and from (3.9),

$$\lim_{k\to\infty} y(k) = 0.$$

Using (3.10) and (2.6), this implies that

$$\lim_{k\to\infty} e(k) = 0.$$

which is the objective of the control scheme.

CONCLUSIONS

An algorithm for discrete time adaptive control of single input-single output systems has been presented. The control input to the plant at any instant of time can be computed on the basis of past samples of the plant. This makes the algorithm particularly attractive for practical implementations, when the computing time must be taken into account.

Moreover, a variable gain in the adaptive algorithm can be used to compensate for the limited range due to finite word length in computing machines.

REFERENCES

[1] T. Ionescu, R. V. Monopoli: "Discrete Model Reference Adaptive Control with an Augmented Error Signal," Automatica, Vol. 13, pp. 507-517, 1977.

[2] L. Hitz, B. D. O. Anderson: "Discrete Positive Real Functions and Their Application to System Stability," Proc. IEE, Vol. 116, No. 1, pp. 153-155, Jan. 1969.

[3] K. S. Narendra, Y. H. Lin: "Stable Discrete Adaptive Control," IEEE TAC, Vol. AC-25, No. 3, pp. 456-461, June 1980.

[4] I. D. Landau, H. M. Silveira: "A Stability Theorem with Applications to Adaptive Control," IEEE TAC, Vol. AC-24, pp. 305-312, April 1979.

[5] G. C. Goodwin, P. J. Ramadge, P. E. Caines: "Discrete Time Multivariable Adaptive Control," IEEE TAC, Vol. AC-25, pp. 449-456, June 1980.

APPENDIX A

Lemma. Let $H(z^{-1}) = J + C[zI-A]^{-1}B$ be an nxn discrete time transfer matrix; then if $\underline{y}(k) = H(z)\underline{u}(k)$ and if H(z) is Strictly Positive Real (SPR) then there exists a matrix $P = P^T$ positive definite, and matrices L, W such that

$$2\underline{u}^T(k)\underline{y}(k) = x^T(k+1)Px(k+1) - x^T(k)Px(k) + ||L^T\underline{x}(k) + Wu(k)||^2 . \tag{A.1}$$

Proof. Let (A,B,C,J) be a minimal realization of H(z),

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + J\underline{u}(k) \end{aligned} \tag{A.2}$$

If H(z) is S.P.R. it is shown in [2] that there exists a matrix $P = P^T$ positive definite, and matrices L, W such that

$$A^TPA - P = -L\ L^T \tag{A.3'}$$

$$A^TPB - C^T = -L\ W \tag{A.3''}$$

$$W^TW = J + J^T - B^T\ P\ B \tag{A.3'''}$$

Now, from (A.3') and (A.2) we get

$$\begin{aligned} -\underline{x}^T(k)LL^T\underline{x}(k) = {} & [\underline{x}^T(k+1) - \underline{u}^T(k)B^T]\ P\ [\underline{x}(k+1) - B\underline{u}(k)] - \\ & - \underline{x}^T(k)P\underline{x}(k) \end{aligned}$$

and then

$$\begin{aligned} -\underline{x}^T(k)LL^T\underline{x}(k) = {} & \underline{x}^T(k+1)P\underline{x}(k-1) + \underline{u}^T(k)B^TPB\underline{u}(k) - 2\underline{x}^T(k+1)PB\underline{u}(k) - \\ & - \underline{x}^T(k)P\underline{x}(k). \end{aligned} \tag{A.5}$$

From (A.3''') we have

$$\underline{u}^T(k)B^TPBu(k) = 2[\underline{y}^T(k) - \underline{x}^T(k)C^T]\ \underline{u}(k) - \underline{u}^T(k)W^TW\underline{u}(k).$$

Finally, from (A.3'') we can write

$$\underline{x}^T(k+1)PB\underline{u}(k) = \underline{x}^T(k)[C^T-LW]\underline{u}(k) + \underline{u}^T(k)B^TPB\underline{u}(k). \tag{A.6}$$

Substituting (A.6) in (A.4), it follows that

$$\begin{aligned} -\underline{x}^T(k)LL^T\underline{x}(k) = {} & \underline{x}^T(k+1)P\underline{x}(k+1) - \underline{x}^T(k)P\underline{x}(k) + 2\underline{x}^T(k)LW\underline{u}(k) - \\ & - 2\underline{y}^T(k)\underline{u}(k) + \underline{u}^T(k)W^TW\underline{u}(k) \end{aligned}$$

from which (A.1) follows.

CONSTRAINED EIGENVALUE/EIGENVECTOR ASSIGNMENT APPLICATION TO FLIGHT CONTROL SYSTEMS

E. Y. SHAPIRO and J. C. CHUNG
Lockheed-California Company
Dept. 72-71, Bldg. 311
Burbank, California 91520

ABSTRACT

The central topic of this paper is the establishment of an efficient practical synthesis procedure for modern flight control systems. Unlike the classical design methodology (Bode plots, Nichols plot, etc.) and optimal control techniques, the present approach provides the designer a direct approach for the synthesis of desired control laws. Although the setting is the now familiar state space, the actual design is performed relative to classical specifications (i.e., modes and mode distribution) by placing closed loop eigenvalues and eigenvectors at some desired locations (regions) within the state space. The new method can handle output feedback configurations subject to controller structural constraints. Complete theoretical background and a realistic numerical example are provided.

I. INTRODUCTION

In flight control systems, an important design constraint is simple and reliable control structure and accurate transient response. This often rules out complex dynamic compensators and/or observers. Thus, constant gain feedback control is highly desirable if it can be achieved.

Recently, output feedback control algorithms have been designed based on quadratic optimal control methods. This approach has its drawbacks, however, since it relies on characterizing design specifications by a scalar cost function. While the results of Harvey and Stein [1] and others partially resolves this issue, a more direct approach which lends itself to interactive design would still be desirable. In this paper, we will offer what we believe is such a methodology.

In the state space setting, it is known that desired system performance is directly related to closed loop eigenvalues and eigenvectors, thus leading to the concept of modal control. Initially, eigenvalue placement via full state feedback was studied and developed. However, the practitioners soon realized that mere eigenvalue placement (or pole placement) is insufficient (because although the proper modes are achieved, the mode distribution can still be rather poor, namely good control of the poles does not imply control over the zeros). Recently, Moore [2] showed that the flexibility offered by full state feedback goes beyond mere eigenvalue placement; as it also offers some degree of eigenvector placement. Thus, true direct modal control can be achieved. Further extension to Moore's work, i.e., eigenvalue/eigenvector placement through output feedback was also studied by Kimura [3] and Srinathkumar [4]. Consequently, at the present time, techniques which are quite competitive with optimal control techniques are available. These techniques can be called direct techniques, as the control law is calculated directly from the desired system performance requirements. These performance requirements must be specified in terms of closed loop eigenvalues and eigenvectors. Close examination of typical aircraft performance requirements as shown in MIL-F-8785B [5] reveals the interesting fact that aircraft performance specifications are in the form of closed loop eigenvalues and eigenvectors. Therefore, modern modal control is a strong contender for the design of advanced flight control systems.

Output feedback, where each output is fed to each input, can still be complex in that there can be many feedback gain elements. We are interested in simplifying the controller by eliminating some of the gains corresponding to accessible outputs.

It is well known that feedback gains do not contribute uniformly to improve total system performance. While many feedback gains can be termed "nonproductive", as they have minimal effect on system performance, a small number of such gains carry the bulk of the effect of improving performance. Consequently, a significant reduction of controller complexity with minimal performance sacrifice becoems feasible. The algorithm developed here includes a systematic procedure for "eliminating" nonproductive gains.

It is the purpose of this paper to develop an efficient and practical synthesis procedure for the design of modern flight control systems, using constant output feedback with gain suppression capability, extending the prior work of Moore, etc. [2, 3, 4, 6]. In this paper we outline the theoretical and the computational aspects of modern modal control techniques for output feedback as well as full state feedback. In addition, a realistic example of a flight control system design will be the subject for the synthesis via modal control.

II. PROBLEM FORMUALTION

Consider the linear time invariant system

$$\dot{x} = Ax + Bu \qquad x(o) = x_o \tag{1}$$
$$y = Cx \tag{2}$$

where $x \epsilon R_n$, $u \epsilon R_m$, $y \epsilon R_r$, and A,B,C are real constant matrices of compatible order. Further assume that the matrices B and C are of full rank, namely

$$\text{rank } [B] = m \tag{3}$$
$$\text{rank } [C] = r \tag{4}$$

Under the above assumptions, the problem considered here can be stated as follows:

Given a self-conjugate set of scalars $\{\lambda_i^d\}$ i = 1,2,. . .,r, a corresponding self-conjugate set of n-vectors $\{\nu_i^d\}$ i = 1,2,. . .,r, and a controller structural constraint in the form of q integer pairs (i.j.), find a real m x r matrix F such that r of the eigenvalues of A + BFC are "close" to the set $\{\lambda_i^d\}$ and the corresponding eigenvectors of A + BFC are "close" to the respective member of $\{\nu_i^d\}$, while the elements of F corresponding to the specified q integer pairs satisfy $f_{i_j} = 0$.

Hence, the sets $\{\lambda_i^d\}$ and $\{\nu_i^d\}$ correspond to the desired closed-loop eigenvalues and eigenvectors, respectively, while the matrix F is a constant output feedback gain matrix of a constrained structure with a corresponding control law

$$u = Fy \tag{5}$$

It was shown [4] that with $y \epsilon R_r$ r-closed loop eigenvalues can be arbitrarily assigned, while simultaneously forcing the respective eigenvectors to approximate some prespecified directions. In practice, quite often, the control system designer is interested mainly in the assignment of only some of the closed loop eigenvalues while allowing the rest to be arbitrarily located. The design specification embedded in the set $\{\nu_i^d\}$ is associated with the mode distribution of the controlled modes. Consequently, a solution to the above problem addresses precisely this commonly encountered practical situation. If more than r modes need to be controlled, then, of course, either more sensors must be added or some type of state estimator (observer) must be employed (dynamic feedback).

III. PROBLEM SOLUTION: FEEDBACK GAIN CALCULATION

Relative to the system (1), (2), and the control law (5), we obtain the closed loop regulator

$$\dot{x} = (A + BFC)\ x \tag{6}$$

for which it has been shown [2, 4] that there exists a real feedback gain matrix F such that

$$(A + BFC)\nu_i = \lambda_i \nu_i \qquad i = 1, 2, \ldots, r \tag{7}$$

where $\{\lambda_i\}$ and $\{\nu_i\}$ are the desired closed loop eigenvalues and eigenvectors respectively. Furthermore, it was also shown [4] that an algebraic expression for F in terms of $\{\lambda_i\}$ and $\{\nu_i\}$ can be obtained in the following manner:

Rearranging (7) we readily obtain

$$(\lambda_i\ I - A)\ \nu_i = BFC\nu_i \tag{8}$$

Since B has full column rank we can assume without loss of generality that

$$B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = \begin{bmatrix} I_m \\ O \end{bmatrix} \tag{9}$$

(If not, a transformation to this form can be found.)

Partitioning (A,B,C) conformably with B we obtain:

$$\left[\begin{array}{c|c} \lambda_i I_m - A_{11} & -A_{12} \\ \hline -A_{21} & \lambda_i I_{n-m} - A_{22} \end{array}\right] \begin{bmatrix} z_i \\ w_i \end{bmatrix} = \begin{bmatrix} I_m \\ O \end{bmatrix} FC \begin{bmatrix} z_i \\ w_i \end{bmatrix} \tag{10}$$

where

$$A = \begin{array}{cc} \begin{array}{cc} m & n-m \end{array} & \\ \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right] & \begin{array}{c} m \\ n-m \end{array} \end{array} \tag{11}$$

$$\nu_i = \begin{array}{cc} \left[\begin{array}{c} z_i \\ \hline w_i \end{array}\right] & \begin{array}{c} m \\ n-m \end{array} \end{array} \tag{12}$$

Considering the first matrix equation of (10), we see that

$$\left[\begin{array}{c|c} \lambda_i I_m - A_{11} & -A_{12} \end{array}\right] \begin{bmatrix} z_i \\ w_i \end{bmatrix} = FC \begin{bmatrix} z_i \\ w_i \end{bmatrix} \tag{13}$$

or equivalently,

$$(A_1 + FC)\nu_i = \lambda_i z_i \tag{14}$$

where the m x n matrix A_1 is given by

$$A_1 = \begin{bmatrix} A_{11} & A_{12} \end{bmatrix} \tag{15}$$

Given a set of r-distinct closed loop eigenvalues (λ_i) and the corresponding set of eigenvectors (ν_i), we can obtain the following equation by repeated use of (14):

$$(A_1 + FC)\ (\nu_1, \nu_2, \nu_3, \ldots \nu_r) = [\lambda_1 z_1 \mid \lambda_2 z_2 \mid \ \cdot\ \cdot\ \cdot\ \cdot\ \mid \lambda_r z_r] \tag{16}$$

To alleviate the need for complex arithmetic when some of the λ_i's are complex, the technique employed in [2] can be used, and with no loss of generality we assume that the necessary transformation [2] has been performed yielding the real matrix equation

$$(A_1 + FC)V = Z_\lambda \tag{17}$$

where

V: Real n x r matrix obtained by transforming $(\nu_1, \nu_2, \ldots, \nu_r)$ according to [2]

Z_λ: Real m x r matrix obtained by transforming $[\lambda_1 z_1 \mid \lambda_2 z_2 \mid \ \cdot\ \cdot\ \cdot \mid \lambda_r z_r]$ according to [2].

In the absence of controller structural constraints, we then get a closed form solution for F from (17), i.e.

$$F = (Z_\lambda - A_1 V)\ (CV)^{-1} \tag{18}$$

The inverse of CV is guaranteed provided the null space of C and the subspace spanned by the columns of V have an empty intersection.

In the presence of controller structural constraints: q elements of the F matrix $(f_{ij1}, f_{ij2}, \ldots, f_{ijq})$ are required to assume the value of zero. In that case, we proceed as follows:

From (17) we obtain

$$F\ (CV) = Z_\lambda - A_1 V \tag{19}$$

denoting

$$\Omega = CV \tag{20}$$

$$\Psi = Z_\lambda - A_1 V \tag{21}$$

we obtain [7]:

$$(I_m \otimes \Omega^T)\ S\ (F) = S(\Psi) \tag{22}$$

Where S(.) is a row stacking operator (i.e., a mapping from the space of m x r matrices to the space of m·r vectors), and $\otimes$ is the Kronecker-Product [7].

When we force certain f_{ij}'s to zero, we can eliminate the corresponding f_{ij}'s from the vector S(F) as well as deleting the corresponding columns in the (m·r) x (m·r) matrix $(I_m \otimes \Omega^T)$. As a result we obtain a set of overdetermined m·r linear equations with m·r-q unknowns of the form

$$W\underline{f} = S(\Psi) \tag{23}$$

where W is the m·r x (m·r - q) matrix obtained from $(I_m \otimes \Omega^T)$ with q columns deleted, and $\underline{f}$ is an m·r-q vector obtained from the vector S(F) by deleting the q f_{ij}'s which are forced to zero.

Consequently, the remaining m·r-q active feedback gains can be computed from

$$\underline{f} = W^{\dagger}\ S(\Psi) \tag{24}$$

where $W^{\dagger}$ is the pseudo-inverse of W. It should be noted that the computation of (24) can be simplified when the following is observed: Equation (22) can be rewritten as

$$\Omega^T\ f_i = \mu_i \tag{25}$$

where f_i is the transpose of the i-th row of F, and μ_i is the transpose of the i-th row of Ψ. Thus, if no element in the i-th row of F is forced to become zero, that row is given by

$$f_i = (\Omega^T)^{-1}\ \mu_i \tag{26}$$

while if some of the elements of the i-th row of F are suppressed to zero, the remaining feedback gains of that row are computed via

$$\underline{f_i} = (\tilde{\Omega}^T)^{\dagger}\ \mu_i \tag{27}$$

where $\underline{f_i}$ are the non-zero gains in the i-th row of F, and $\tilde{\Omega}^T$ is obtained from Ω^T by deleting the columns corresponding to the gains which are suppressed to zero.

We note that (18) and (26) and (27) yield a solution for F provided r closed loop eigenvaleus and their respective eigenvectors are known. Although we are able to insert the set $\{\lambda_i^d\}$ i = 1, 2, ..., r, our desired closed loop eigenvalues into (18), via (16), we are not at liberty to replace $\nu_1, \nu_2, \ldots, \nu_r$ by our desired set of closed loop eigenvectors $\{\nu_i^d\}$ i = 1, 2, ..., r, since the set $\{\nu_i^d\}$ might not belong to the set of assignable closed loop eigenvectors. It will, therefore, be necessary to find some approximation to the set $\{\nu_i^d\}$ which is assignable, yet "close" in some sense to the original $\{\nu_i^d\}$'s, and only then proceed to compute F from either (18) or (26) and (27).

IV. PROBLEM SOLUTION: EIGENVECTOR APPROXIMATION

In order to assign $\{\lambda_i^d\}$ as closed loop eigenvalues as well as assigning an approximation to $\{\lambda_i^d\}$ as the respective closed loop eigenvectors, one may rewrite (8) in the following form [2]:

$$\nu_i = (\lambda_i I - A)^{-1}\ BFC\nu_i \tag{28}$$

and define the m-component vector m_i by

$$m_i = FC\nu_i \qquad i = 1, 2, \ldots, r \tag{29}$$

Then we obtain [2]

$$\nu_i = (\lambda_i I - A)^{-1} B\ m_i \tag{30}$$

The implication of (30) is that the closed loop eigenvector ν_i corresponding to the closed loop eigenvalue λ_i must lie in the subspace spanned by the columns of the matrix $(\lambda_i I-A)^{-1}B$. Therefore, the designer has the freedom to arbitrarily select a set of r m-vectors (r = rank [C], m = rank [B]) such that the closed loop control system will have the desired performance characteristics. The only restriction on m_j is that $m_i = m_j^*$ whenever $\lambda_i = \lambda_j^*$. This condition must be satisfied in order to obtain a real feedback matrix [2].

In order to capitalize on the freedom in the assignment of the vectors $\{m_i\}$, it is beneficial to select the $\{m_i\}$ such that the left-hand side of (30) best approximates the desired eigenvector ν_i^d. Therefore, we define a cost function of the form

$$J(m_i) \triangleq (\nu_i^d - \nu_i)^T(\nu_i^d - \nu_i) = (\nu_i^d - L_i m_i)^T(\nu_i^d - L_i m_i) \tag{31}$$

where

$$L_i \triangleq (\lambda_i I - A)^{-1}B \tag{32}$$

and proceed to minimize this cost function with respect to m_i.

$$\frac{d}{dm_i}\left\{ J(m_i) \right\} = \frac{d}{dm_i}\left[(\nu_i^d - L_i m_i)^T(\nu_i^d - L_i m_i) \right] = 2L_i^T(L_i m_i - \nu_i^d) \tag{33}$$

By setting (33) to zero, we obtain the following "optimal" choice for the vector m_i

$$m_i = (L_i^T L_i)^{-1} L_i^T \nu_i^d \tag{34}$$

Obviously due to ill conditioning of $(L_i^T L_i)$, special care from a numerical standpoint must be exercised when evaluating m_i from (34).

Substituting the m_i's obtained from (34) into equation (30) yields a set of optimum closed loop eigenvectors which best approximate the desired eigenvectors (ν_i^d) in the least squares sense.

However, it will generally be true that only a few components in the desired eigenvector are actually specified; while the rest can be arbitrary. In this case, we will use a method introduced by Harvey and Stein [1].

Consider the desired closed loop eigenvector

$$\nu_i^d = \begin{bmatrix} \nu_{1i} \\ x \\ \nu_{3i} \\ x \\ x \\ \nu_{ni} \end{bmatrix} \tag{35}$$

where

ν_{ji} = designer specified components

x = don't care (unspecified) components

We will introduce a row reordering operator [1], $\{\ \}^{R_i}$. The operation involves simultaneous row reordering of the matrix $\{(\lambda_i I - A)^{-1}B\}$ and the ν_i^d vector until ν_i^d is partitioned into two subvectors. One contains the specified components and the other contains the unspecified components. After the row reordering operation, we have the following matrix representation.

$$\left\{ (\lambda_i I - A)^{-1}B \right\}^{R_i} = \begin{bmatrix} \tilde{L}_i \\ D_i \end{bmatrix} \tag{36}$$

$$\left\{ \nu_i^d \right\}^{R_i} = \begin{bmatrix} \ell_i \\ \hline d_i \end{bmatrix} \tag{37}$$

where ℓ_i = contains the specified components of ν_i^d

d_i = contains the unspecified components of ν_i^d

Employing the same least square minimizatior procedure, we obtain the following result.

$$m_i = (\tilde{L}_i^T \tilde{L}_i)^{-1} \tilde{L}_i^T \ell_i \tag{38}$$

Thus, using the set $\{m_i\}$ obtained either from (34) or from (38), we can calculate the set of r achievable closed loop eigenvectors $\{\nu_i\}$ from (30). Once we have obtained the r-closed loop assignable eigenvectors, the feedback gain matrix F can be determined according to (18) for the unconstrained controller or from (26) and (27) for the constrained controller.

V. A NUMERICAL EXAMPLE

Consider the SAS (Stability Augmentation System) design for the lateral axis, rigid body model of the L-1011 aircraft at cruise flight condition.

The linearized perturbation equations are given by

$$\dot{x} = Ax + Bu \tag{39}$$

$$y = Cx \tag{40}$$

where A, B, and C are

$$A = \begin{bmatrix} -20 & 0 & 0 & 0 & 0 & 0 \\ 0 & -25 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ -.744 & -.032 & 0 & -.154 & -.0042 & 1.54 \\ .337 & -1.12 & 0 & .249 & -1.00 & -5.20 \\ .02 & 0 & .0386 & -.996 & -.000295 & -.117 \end{bmatrix} \tag{41}$$

$$B = \begin{bmatrix} 20 & 0 \\ 0 & 25 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{42}$$

$$C = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{43}$$

The state vector is given by

$$x = \begin{bmatrix} \delta_r \\ \delta_a \\ \phi \\ r \\ p \\ \beta \end{bmatrix} \begin{matrix} \text{- rudder deflection (radians)} \\ \text{- aileron deflection (radians)} \\ \text{- bank angle (radian)} \\ \text{- yaw rate (rad/sec)} \\ \text{- roll rate (rad/sec)} \\ \text{- sideslip angle (radians)} \end{matrix}$$

and the control vector u is given by

$$u = \begin{bmatrix} \delta_{r_c} \\ \delta_{a_c} \end{bmatrix} \begin{matrix} \text{- rudder command (radians)} \\ \text{- aileron command (radians)} \end{matrix}$$

The free aircraft eigenvalues (i.e. the eigenvalues of the A matrix) are:

$$\left.\begin{array}{l} \lambda_1 = -20 \text{ - rudder actuator mode} \\ \lambda_2 = -25 \text{ - aileron actuator mode} \\ \left.\begin{array}{l} \lambda_3 = -.0884 + j\ 1.272 \\ \lambda_4 = -.0884 - j\ 1.272 \end{array}\right\} \text{ dutch roll mode} \\ \lambda_5 = -.1.085 \text{ - roll subsidence mode} \\ \lambda_6 = -.00911 \text{ - spiral mode} \end{array}\right\} \tag{44}$$

The dutch roll (λ_3 & λ_4) frequency and damping are $\omega = 1.275$ $\zeta = .0693$. For the purpose of SAS design we usually feedback body rates (r and p) as well as the sideslip angle β (lateral acceleration is often used instead of β). The purpose of the SAS is to properly position the dutch roll mode and the roll subsidence mode, and furthermore to decouple the dutch roll (a yaw rate and sideslip motion) from the roll mode. For the purpose of this example, we shall assume that the desired dutch roll and the roll subsidence modes are:

$$\lambda^d_{1,2} = -1.50 \pm j\ 1.50 \tag{45}$$

$$\lambda^d_3 = -2.00 \tag{46}$$

To reflect the desired decoupling, the corresponding desired eigenvectors are (using the notation of (35)).

$$\underbrace{\nu^d_1 = \begin{bmatrix} x \\ x \\ 0 \\ 1 \\ 0 \\ x \end{bmatrix} \qquad \nu^d_2 = \begin{bmatrix} x \\ x \\ 0 \\ x \\ 0 \\ 1 \end{bmatrix}}_{\text{dutch roll}} \qquad \nu^d_3 = \underbrace{\begin{bmatrix} x \\ x \\ x \\ 0 \\ 1 \\ 0 \end{bmatrix}}_{\text{roll subsidence}} \tag{47}$$

Using the procedures outlined above, several output feedback matrices corresponding to different controller structures were computed; the results are summarized in Table 1.

CONFIGURATION	FEEDBACK GAIN MATRIX F	CLOSED LOOP EIGENVALUES $\lambda_i(A - BFC)$				
		RUDDER MODE	AILERON MODE	DUTCH ROLL	ROLL SUBSIDENCE	SPIRAL MODE
I	$\begin{array}{c|ccc} & r & p & \beta \\ \hline \delta_{r_c} & -3.18 & -0.0051 & 2.67 \\ \delta_{a_c} & -1.35 & -0.826 & 5.42 \end{array}$	-17.30	-23.96	$-1.50 \pm j\ 1.50$ $\omega = 2.12$ $\zeta = 0.707$	-2.000	-0.0030
II	$\begin{bmatrix} -3.18 & 0 & 2.67 \\ 0 & -0.804 & 3.51 \end{bmatrix}$	-17.33	-24.02	$-1.456 \pm j\ 1.522$ $\omega = 2.10$ $\zeta = 0.691$	-2.004	-0.0034
III	$\begin{bmatrix} -3.18 & 0 & 2.67 \\ 0 & -0.774 & 0 \end{bmatrix}$	-17.33	-24.06	$-1.416 \pm j\ 1.449$ $\omega = 2.02$ $\zeta = 0.698$	-1.995	-0.06

Table 1: Feedback Gains and Corresponding Closed Loop Poles

Note the precise pole placement on the dutch roll mode and the roll subsidence mode (compared with (45) and (46)), while the rest of the closed loop eigenvalues remain close to the positions they assumed in the open loop.

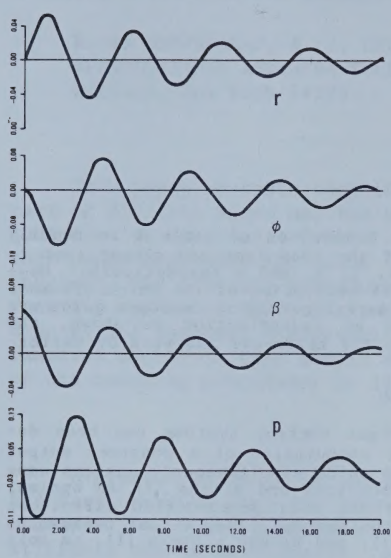


Figure 1: Open Loop Response to Initial Sideslip Angle of .05 [rad]

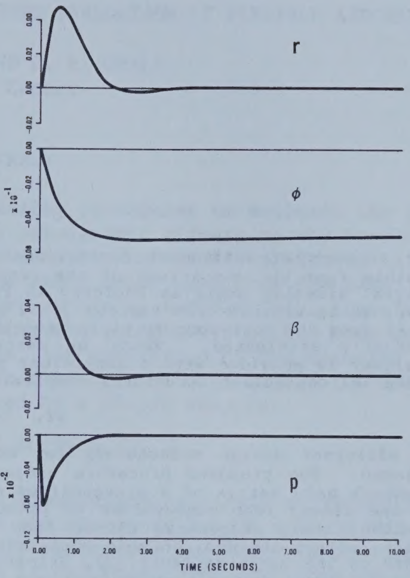


Figure 2: Closed Loop Response to Initial Sideslip Angle of .05 [rad]

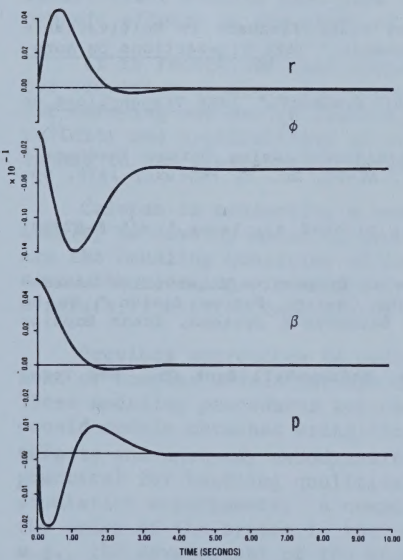


Figure 3: Closed Loop Response to Initial Sideslip Angle of .05 [rad] for Controller Structure Case II

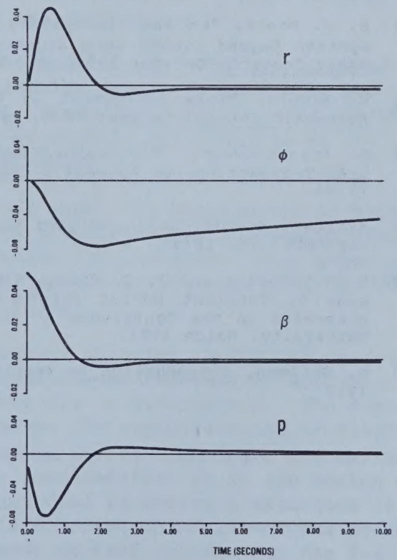


Figure 4: Closed Loop To Initial Sideslip Angle of .05 [rad] for Controller Structure Case III

The eigenvector assignment accomplished by the F matrices of Table 1 is readily visible from the comparison of the responses of the open loop and closed loop to initial sideslip angle as depicted in Figures 1, 2, 3, and 4 respectively. However, it is visible from Figures 2, 3, and 4 that decoupling of the dutchroll mode (r,β) from the roll-mode (p,ϕ) is progressively deteriorating as feedback gains are gradually eliminated. Thus, by carrying out an investigation as shown, the designer is provided with a convenient procedure for trade-off analysis of performance vs. controller structural complexity.

VI. CONCLUSION

An efficient design methodology for modern flight control systems has been developed. The proposed procedure hinges on the computation of a constant output feedback gain matrix of a prespecified structure which simultaneously assigns some of the closed loop eigenvalues to prescribed locations and aligns (in an optimal fashion) their respective closed loop eigenvectors with prespecified directions which reflect the desired mode distribution in the system. Theoretical background based on the works of Moore [2], Srinathkumar [3], and Harvey & Stein [1], as well as a realistic design problem, has been presented.

REFERENCES

[1] C. A. Harvey and G. Stein, "Quadratic Weights for Asymptotic Regulator Properties," IEEE Transactions on Automatic Control, Vol. AC-23, No. 3, June 1978, pp. 278-387.

[2] B. C. Moore, "On the Flexibility Offered by State Feedback in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment," IEEE Transactions on Automatic Control, October 1976, pp. 689-692.

[3] H. Kimura, "Pole Assignment by Gain Output Feedback," IEEE Transactions on Automatic Control, August 1975, pp. 509-516.

[4] S. Srinathkumar, "Eigenvalue/Eigenvector Assignment Using Output Feedback," IEEE Transactions on Automatic Control, Vol. AC-23, No. 1, February 1978, pp. 79-81.

[5] Military Specification "Flying Qualities of Piloted Airplanes," MIL-F-8775B, September 16, 1974.

[6] E. Y. Shapiro and J. C. Chung, "Application of Eigenvalue/Eigenvector Assignment by Constant Output Feedback to Flight Control System Design," To be presented in the Conference on Information Sciences & Systems, Johns Hopkins University, March 1981.

[7] R. Bellman, Introduction to Matrix Analysis, McGraw-Hill Book Co., New York, 1970

MODELING PROCEDURES FOR HANDLING QUALITIES EVALUATION OF FLEXIBLE AIRCRAFT*

K. S. GOVINDARAJ, B. J. EULRICH AND C. R. CHALK
Arvin/Calspan Advanced Technology Center
Buffalo, New York 14225

ABSTRACT

This paper presents simplified modeling procedures to evaluate the impact of flexible modes and the unsteady aerodynamic effects on the handling qualities of Supersonic Cruise Aircraft (SCR). The results presented in this paper represent the first task of a study being conducted for NASA Langley. The modeling procedures involve obtaining reduced order transfer function models of SCR vehicles, including the important flexible mode responses and unsteady aerodynamic effects, and conversion of the transfer function models to time domain equations for use in simulations. The use of the modeling procedures is illustrated by a simple example.

1. INTRODUCTION

A gradual increase in the size and flexibility of transport aircraft has been observed over the years. With the advent of active control for maneuver load, gust load, and flutter alleviation, a further reduction in stiffness can be anticipated. Although it has been recognized that greater flexibility will also influence flight stability and control, handling qualities, and ride qualities, there has been insufficient exploration of these effects to determine whether or not they will have a significant impact on the development of Supersonic Cruise Aircraft (SCR). Also, insufficient results have been obtained to determine the significance of dynamic effects in the study of handling and ride qualities.

It is recognized that simplified time domain equations are required for simulator studies, for formulating handling qualities criteria, and for carrying out design studies. However, questions remain concerning the validity and applicability of reduced order models, particularly in response to the treatment of unsteady aerodynamics.

Calspan is conducting a research study under the sponsorship of NASA Langley to develop modeling procedures and simulation experiments to evaluate the handling qualities of large, flexible SCR aircraft. This paper presents the modeling procedures for handling qualities evaluation of the SCR aircraft, the first task of the research study.

Previous approaches to modeling have been directed towards the development of complex models for use in structural design and flutter analysis. These modeling procedures are described briefly in Reference 1. The time domain models obtained using these approaches for representation of flexible effects and unsteady aerodynamics are complex and of high order and are impractical for handling qualities criteria consideration or in the design of simulation experiments. A common technique used to obtain a reduction in the order of the system is the quasistatic elimination of structural modes, e.g., the development of the structural mode control systems for the B-1 aircraft (Reference 2). However, this technique has a shortcoming in that it also requires approximations of the unsteady aerodynamics. Thus, errors of unknown magnitude are introduced into the reduced order equations.

---

*This work was supported by the NASA Langley Research Center under Air Force Contract No. F33615-79-C-3618.

In the past few years, Calspan has developed methods for identifying the dynamic characteristics of an aircraft from flight measurements in order to obtain a valid model for control studies. Recently, a method was developed (Reference 3) for obtaining a model which is valid for the frequency range import for handling and ride qualities. This technique identifies the actual system including unsteady aerodynamic effects with models of appropriate order for the required frequency range. The method involves fitting transfer functions to frequency response data measured in flight and conversion of the transfer functions to time domain equations. The modeling procedure for handling qualities evaluation of SCR vehicles is based on this method and consists of two steps:

1. Obtain reduced order transfer function models of SCR vehicles including the important flexible mode responses and the unsteady aerodynamic effects from the frequency response data.
2. Convert the transfer function models obtained in step (1) to time domain equations of motion or, equivalently, to a state space representation.

The transfer function estimation method is the same as the method of Reference 3. The method for converting the transfer function models to time domain equations is an alternative to the method of Reference 3 with improved computational qualities.

The remainder of the paper is organized as follows: Section 2 presents the transfer function estimation method. Section 3 presents the method for converting the transfer function models to state space representation. An example to illustrate the use of the modeling procedures is given in Section 4. Concluding remarks are given in Section 5.

2. TRANSFER FUNCTION ESTIMATION

The first step in the development of SCR models is to estimate transfer functions from frequency response data representing the effects of flexible modes and unsteady aerodynamics. The transfer functions are estimated from the available amplitude and phase data at discrete frequencies. A least squares technique (Reference 3) is used to fit transfer functions to the frequency response data. This is a classical nonlinear regression problem which is generally a nontrivial problem to solve for high order systems. The dimensionality problem can be reduced by recognizing that the denominator polynomial is the same for each transfer function to be identified.

The least squares problem will be simplified if the denominator polynomial is known. For this case the estimation problem will be linear and the numerator polynomial for each transfer function can be determined separately in a straightforward manner.

The transfer function model to be estimated is expressed as the ratio of two frequency-dependent polynomials

$$G(j\omega_k) = \frac{N(j\omega_k)}{D(j\omega_k)} = \frac{b_o+b_1(j\omega_k)+b_2(j\omega_k)^2+\ .....}{1+d_1(j\omega_k)+d_2(j\omega_k)^2+\ .....} = \hat{R}(k,\theta)+j\ \hat{I}\ (k,\theta) \quad (1)$$

where $\hat{R}(k,\theta)$ and $\hat{I}(k,\theta)$ are the real and imaginary parts of the model evaluated at the discrete frequency ω_k and $\theta = (b_o,b_1,\ldots d_1,d_2\ldots)^T$, the parameter vector to be estimated.

The complex frequency response data from which the parameters of the model, θ, are to be estimated are expressed in terms of real and imaginary parts at r discrete frequencies as

$$F(j\omega_k) = R(k) + j\,I(k),\ k = 1,\ 2,\ \ldots\ r \tag{2}$$

It is assumed that measurement accuracies of $R(k)$ and $I(k)$ are known with equal variances of $\sigma^2(k)$.

The weighted least-squares solution is obtained by minimizing the cost functional,

$$J(\theta) = \frac{1}{2}\sum_{k=1}^{r}|\varepsilon(k)|^2\frac{1}{\sigma^2(k)} = \frac{1}{2}\sum_{k=1}^{r}\{[R(k)-\hat{R}(k,\theta)]^2+[I(k)-\hat{I}(k,\theta)]^2\}\frac{1}{\sigma^2(k)} \tag{3}$$

with respect to θ, where $\varepsilon(k) = F(j\omega_k) - G(j\omega_k)$, the error at discrete frequency ω_k. The solution to this minimization problem is given by the root of $\partial J(\theta)/\partial\theta = 0$. Since $\hat{R}(k,\theta)$ and $\hat{I}(k,\theta)$ are nonlinear in θ, a nonlinear programming technique and an initial parameter estimate are required to start the search for a solution. If the denominator coefficients $(d_1,\ d_2,\ \ldots.)$ are known, then $\hat{R}(k,\theta)$ and $\hat{I}(k,\theta)$ become linear functions of the remaining unknown numerator coefficients in θ. A simple closed form solution can be obtained for the minimization problem under these conditions. An iterative technique is given in Reference 3 which simplifies the problem of minimization of the cost functional of Equation (3). Identification of multiple transfer functions using this technique is straightforward. This technique has been used successfully for the identification of transfer functions from flight data (Reference 3).

3. TIME DOMAIN MODEL DEVELOPMENT

The second step in the development of SCR models is to convert the transfer functions estimated from frequency response data to time domain equations of motion or, equivalently, to a state space represention.

The desired time-domain representation of the flexible equations of motion of SCR aircraft is given by the following state-space equations:

$$\begin{aligned}\dot{x} &= Fx + Gu\\ y &= Hx\end{aligned} \tag{4}$$

where x is a vector of dimension n representing the vehicle rigid body states q, α, etc. and the normal mode variables η and derivatives $\dot{\eta}$, u is a vector of dimension m representing the control inputs and y is an output vector of dimension ℓ representing the vehicle sensor outputs, e.g. q, α, and n_x, n_y, n_z.

The transfer function matrix of the system of Equation (4) is given by

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_\ell \end{bmatrix} = \frac{1}{D(s)} \begin{bmatrix} N_{11} & N_{12} & \cdots & N_{1m} \\ N_{21} & N_{22} & & N_{2m} \\ \vdots & \vdots & & \vdots \\ N_{\ell 1} & N_{\ell 2} & \cdots & N_{\ell m} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} \tag{5}$$

where $D(s)$ is the characteristic polynomial and N_{ij} are the numerator polynomials.

For this study, the input data are the amplitude and phase as a function of frequency of the transfer functions of Equation (5). The first step is to fit the frequency response data to obtain the transfer function matrix of Equation (5) as described in the previous section. The next step is to convert the transfer function matrix to time-domain equations of the form of Equation (4). It is generally not possible to convert the transfer function matrix of Equation (5) directly to the time-domain equations of the form of Equation (4) since the elements of F, G and H are nonlinear functions of the coefficients of the transfer functions. Hence, the transfer functions are first converted to the phase variable form of time-domain equations and then transformed to the form of Equation (4). Time-domain equations in phase variable form for a single input case can be determined in a straightforward manner (Reference 3) from the output transfer functions. For example, the output transfer functions with respect to the ith input estimated from the frequency response data are given by

$$\frac{Y}{U_i}(s) = \frac{1}{D(s)} \begin{bmatrix} N_{1i}(s) \\ N_{2i}(s) \\ N_{3i}(s) \\ \vdots \\ N_{\ell i}(s) \end{bmatrix} \tag{6}$$

where

$$\begin{aligned} N_{1i}(s) &= h^0_{1i} + h^1_{1i}s + h^2_{1i}s^2 + \dots\, h^{n-1}_{1i}s^{n-1} \\ N_{2i}(s) &= h^0_{2i} + h^1_{2i}s + h^2_{2i}s^2 + \dots\, h^{n-1}_{2i}s^{n-1} \\ &\vdots \\ N_{\ell i}(s) &= h^0_{\ell i} + h^1_{\ell i}s + h^2_{\ell i}s^2 + \dots\, h^{n-1}_{\ell i}s^{n-1} \end{aligned} \tag{7}$$

The time domain equations in phase variable form are given as

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \vdots \\ \dot{z}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & \\ -d_0 & -d_1 & \cdots & & \cdots & -d_{n-1} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u_i = F_o z + G_o u_i$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_\ell \end{bmatrix} = \begin{bmatrix} h_{1i}^{n-1} & h_{1i}^{n-2} & \cdots\cdots & h_{1i} \\ h_{2i}^{n-1} & h_{2i}^{n-2} & \cdots\cdots & h_{2i}^{0} \\ \vdots & & & \\ h_{\ell i}^{n-1} & h_{\ell i}^{n-2} & \cdots\cdots & h_{\ell i}^{0} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = H_i z \qquad (8)$$

The phase variable time-domain equations can be transformed to the form of Equation (4) if the phase variable transformation matrix T_i is available. For the case where all the states are measured directly, i.e. $y = x$, the transformation matrix is formed from the numerator polynomial coefficients of the transfer functions. Each row of the transformation matrix is determined from a particular numerator polynomial coefficient. The output equation is given as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} t_{1i}^{\;n-1} & t_{1i}^{\;n-2} & \cdots\cdots & t_{1i}^{\;0} \\ t_{2i}^{\;n-1} & t_{2i}^{\;n-2} & \cdots\cdots & t_{2i}^{\;0} \\ \vdots & & & \\ t_{ni}^{\;n-1} & t_{ni}^{\;n-2} & \cdots\cdots & t_{ni}^{\;0} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = T_i z \qquad (9)$$

$$N_{1i} = t_{1i}^{\;0} + t_{1i}^{\;1} s + t_{1i}^{\;2} s^2 + \ldots\ldots \; t_{1i}^{\;n-1} s^{n-1}$$

$$\vdots$$

$$N_{ni} = t_{n1}^{\;0} + t_{ni}^{\;1} s + t_{ni}^{\;2} s^2 + \ldots\ldots \; t_{ni}^{\;n-1} s^{n-1}$$

The time domain equations in terms of the x state variables are given by

$$\dot{x} = Fx + G_i\, u_i \quad ; \quad F = T_i\, F_o\, T_i^{-1} \quad G_i = T_i\, G_o \qquad (10)$$

where G_i is the ith column of the input matrix . The input-output transfer function relationships are the same for both the phase variable form of Equation (9) and the form of Equation (10) regardless of the accuracy of the original transfer functions.

Development of time-domain equations from the transfer function matrix of Equation (5) is more involved. For each set of output transfer functions, phase variable time-domain equations of the form of Equation (8) can be obtained. If the output vector y is the same as the state vector x, transformation from phase-variable form to the state-space form of Equation (4) is accomplished simply by the relationships

$$F = T_i\, F_o\, T_i^{-1} \quad ; \quad G_i = T_i^{-1}\, G_o \qquad i=1,\, 2,\, \ldots m$$

As an example, for a two-input system, two separate transformation matrices, T_1 and T_2, can be defined and each transformation must lead to the same system matrix, i.e.

$$F = T_1\, F_o\, T_1^{-1} = T_2\, F_o\, T_2^{-1} \qquad (11)$$

or, after a little algebra, the following relationship is obtained

$$F_o T_1^{-1} T_2 = T_1^{-1} T_2 F_o \tag{12}$$

This equation represents a constraint that must be satisfied to preserve the input-output transfer function relationships. Specifically, it is a physical constraint that must be imposed to ensure that the system matrix, F, obtained from each phase variable transformation came from the same physical system. For single-input transformations, the input-output transfer function relationships are preserved regardless of the accuracy of the original transfer functions. For a multiple input system, if the constraint of Equation (12) is not imposed directly as part of the transfer function fitting process, it will never be satisfied exactly unless the fit to the data is absolutely perfect. Hence, any irreducible state-space representation of the transfer function matrix that preserves its input-output relationships would generally be of higher order than the original transfer function model — a seemingly undesirable feature. Accordingly, these constraints formed the fundamental basis for the accuracy verification concepts described in Reference 3 as well as the proposed procedure for conversion of the transfer function matrix model to a state-space representation.

In effect, the method of Reference 3 forces these constraints to be satisfied after the fitting process. The resulting state-space model would be of order equal to the original transfer function model and perhaps may be an indirect way to improve upon the accuracy of the resulting model. The major drawbacks of this method are that: (1) the number of measurements or outputs must be equal to the system order; (2) there is no guarantee that any of the original input-output relationships are preserved; and (3) at present, there is no clear cut way to establish criteria or guidelines which can be employed to make the method straightforward to use in practice.

These drawbacks can be eliminated by directly satisfying the constraints as part of the fitting process. One approach is to directly estimate the elements of the F and G matrices of the state-space model of Equation (4) instead of the coefficients of transfer function polynomials from the frequency response data. Since a state-space model is obtained, the conversion of the transfer function matrix model to a state-space model would not be required. However, for a higher order system, the computational requirement with this approach will be prohibitive.

For this study, the simplicity of the two-step approach to model development stated in the introduction is too good to ignore. Step (1) involving the transfer function identification from frequency response data is straightforward. However, a simplified approach to convert the resulting transfer function matrix model to a state-space representation for step (2) is still required. The remainder of this section is devoted to the description of this simplified approach.

The linearized flexible equations of motion of a SCR aircraft for two inputs is given in the state-space form by

$$\dot{x} = Fx + G_1 u_1 + G_2 u_2 \quad ; \quad y = Hx \tag{13}$$

A similarity transformation matrix T_1 for the input u_1 can be found to transform the system into the phase variable form

$$\dot{z} = F_o z + G_o u_1 + G_{o2} u_2 \; ; \; y = H_1 z$$

where $$x = T_1 z \; ; \; H_1 = HT_1 \; ; \; G_o = T_1^{-1} G_1 \; ; \; G_{o2} = T_1^{-1} G_2 \qquad (14)$$

and F_o and G_o are as defined by Equation (8). The matrix G_{o2} does not have a general form. The transfer function matrix model can now be written as

$$Y(s) = \frac{H_1 adj\ [sI-F_o]G_o}{D(s)} U_1(s) + \frac{H_1 adj\ [sI-F_o]G_{o2}}{D(s)} U_2(s)$$

$$= \frac{H_1 S}{D(s)} U_1 + \frac{H_1 adj\ [sI-F_o]G_{o2}}{D(s)} U_2(s) \; ; \; S = [1 \;\; s \;\; s^2 \ldots s^{n-1}]^T \qquad (15)$$

As stated previously, the elements of H_1 are the numerator polynomial coefficients for the transfer function model relating the ℓ outputs to the u_1 input.

The transfer function matrix model estimated from the frequency response data fitting process can be written as

$$[Y_1(s) \; Y_2(s) \; \ldots \; Y_\ell(s)]^T = \frac{1}{D(s)} \begin{bmatrix} N_{11}(s) & N_{21}(s)..N_{\ell 1}(s) \\ N_{12}(s) & N_{22}(s)..N_{\ell 2}(s) \end{bmatrix}^T [U_1(s) \; U_2(s)]^T \qquad (16)$$

The objective is to convert the transfer function matrix model of Equation (16) into a phase variable time-domain equation of the form of Equation (14) and, if the transformation matrix T_1 is available, to transform to the F and G form of Equation (13). The time-domain model development approach is to satisfy the constraints of Equation (12) by forcing the state space model to conform to the one given by Equation (14). This can be done simply by equating the coefficients of like powers of s in Equation (15) and (16) and solving for the unknowns, H_1 and G_{o2}. Equating the coefficients of like powers of s yields the following set of linear equations:

$$H_1 S = [N_{11}(s) \; N_{21}(s) \; .. \; N_{\ell 1}(s)]^T$$

$$H_1 \; adj\ [sI-F_o]G_{o2} = [N_{12}(s) \; N_{22}(s) \; .. \; N_{\ell 2}(s)]^T \qquad (17)$$

The resulting state-space representation is given by Equation (14). As shown by Equation (8), the matrix H_1 is directly determined from the Equation (17). With F_o and H_1 known, the second equation of Equation (17) is a linear set of equations in the n unknown elements of G_{o2}. This result can be shown as follows. The known matrix of polynomials $H_1 \; adj\ [sI-F_o]$, can be written as

$$H_1 adj[sI-F_o] = \begin{bmatrix} b_{11}^o + b_{11}^1 s + \ldots . b_{11}^{n-1} \; \ldots . . b_{1n}^o + b_{1n}^1 s + \ldots b_{1n}^{n-1} s^{n-1} \\ b_1^o + b_1^1 s + \ldots . b_1^{n-1} s^{n-1} \; \ldots b_n^o + b_n^1 s + .. b_n^{n-1} s^{n-1} \end{bmatrix} \qquad (18)$$

The unknown elements of G_{o2} and the known output transfer functions with respect to the second input are defined as

$$G_{o2} = [g_{o12} .. g_{on2}]^T ; \; [N_{12}(s) .. N_{\ell 2}(s)]^T = [h_{12}^o + .. h_{12}^{n-1} s^{n-1} .. h_{\ell 2}^o + .. h_{12}^{n-1} s^{n-1}]^T \qquad (19)$$

From Equation (17), by equating powers of s, the following ℓ sets of linear equations in n unknowns are obtained

$$\begin{bmatrix} b^{o}_{i1} & b^{o}_{i2} & \dots b^{o}_{in} \\ b^{n-1}_{i1} & b^{n-1}_{i2} & \dots b^{n-1}_{in} \end{bmatrix} \begin{bmatrix} g_{o12} \\ g_{on2} \end{bmatrix} = \begin{bmatrix} h^{o}_{i2} \\ h^{n-1}_{i2} \end{bmatrix} \quad i = 1,2,..\ell \tag{20}$$

Theoretically, under certain observability conditions, if the transfer function fitting process was perfect, each set of equations would provide the same G_{o2}. Practically, this is obviously not the case so that a least squares solution for G_{o2} is more desirable. The system is completely observable if there is no pole-zero cancellation. If there is a pole-zero cancellation in one of the transfer functions with respect to the second input, the set of linear equations obtained from this transfer function would be linearly dependent. The remaining linearly independent set of equations can still be used to provide a solution for G_{o2}.

The same process can be repeated to determine the input matrices from transfer functions for more than two inputs. The resulting state-space model is then given by

$$\dot{z} = F_o z + G_o u_1 + G_{o2} u_2 + G_{o3} u_3 + .. G_{om} u_m \;;\; y = H_1 z \tag{21}$$

As stated previously, if all the states are measured directly, the transformation matrix, T_1, to transform to the form of Equation (4) is available and the time-domain equations are given by

$$\dot{x} = Fx + G_1 u_1 + G_2 u_2 + \dots G_m u_m \;;\; y = Hx \tag{22}$$

$$F = T_1 F_o T_1^{-1} \;;\; H = H_1 T_1^{-1} \;;\; G_1 = T_1 G_o \;;\; G_1 = T_1 G_{oi},\; i=2, \dots m \tag{23}$$

In using this method for time domain model development, a few comments and concerns are appropriate:

- The method is simple and easy to use in practice, but it is only an approximation. However, it does not have the computational problems of the method proposed in Reference 3.
- The results are dependent upon the H_1 matrix or, more appropriately, the control input used for the similarity transformation. However, there is freedom of selection and appropriate measures are available upon which to base a selection. Controllability and observability measures can be examined using the original estimated transfer function matrix model to provide a basis for selection (Reference 3). Measures such as rms fit error or variance estimates can also be used.
- For higher order systems, the computation of $adj[sI-F_o]$ in Equation (17) would be very involved. A computationally trivial algorithm has been developed which makes use of the special form of F_o to compute $adj[sI-F_o]$. Due to space limitation, it will not be presented in this paper.
- The time-domain model of Equation (21) represents the small perturbation equations of motion of the SCR aircraft about a particular flight condition. This form of the equations is inadequate to describe large angle motions about this flight condition. However, if enough transfer functions are available to transform to the form of Equation (22), inclusion of additional terms into the time-domain equations to account for large angle maneuvers would be feasible.

As stated previously, a least squares solution is more appropriate in practice. It is computed by combining the linear sets of equations to give the linear regression model

$$B = gh; \; B = [B_1^T \; B_2^T \ldots B_\ell^T]; \; h = [h_1^T \; h_2^T \ldots . h_\ell^T]^T; \; g = [g_{o22} \cdots \; g_{on2}]^T \tag{25}$$

where B_i and h_i are formed from the linear sets of Equation (20). The solution is given as

$$\hat{g} = (B^T R^{-1} B)^{-1} B^T R^{-1} h \tag{26}$$

where R is a diagonal weighting matrix. It can be shown that the estimate $\hat{g}$ is unbiased whether obtained as the solution of linear sets of Equations (20) or as the least squares solution.

4. EXAMPLE

This section presents an example to illustrate the use of the mod ling procedures described in the previous sections. Frequency response data were generated from a known system given by the equations

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ -10 & -3 & 0 \\ 0 & 2 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 10 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \; ; y = x \tag{27}$$

The frequency response data consists of the amplitude and phase of the transfer functions of the system of Equation (27):

$$Y(s) = \frac{1}{s^3+6s^2+9s+20} \begin{bmatrix} 20 & s^2+6s+9 \\ 10s^2+30s & -10s-30 \\ 20s & -20 \end{bmatrix} U(s) \tag{28}$$

The modeling procedures will be used to determine the time-domain equations (27) from the frequency response data. The first step is to estimate the transfer functions from the amplitude and phase data using techniques described in Section 2. The estimated transfer functions are given by

$$Y(s) = \frac{1}{s^3+6s^2+9s+20} \begin{bmatrix} -3.51x10^{-6} s^2+1.96x10^{-5} s+20 & s^2+6s+9 \\ 10s^2+30s-1.27x10^{-4} & -6.32x10^{-6} s -10s-30 \\ 5.4x10^{-6} s^2+20s-1.14x10^{-4} & 3.51x10^{-6} s^2 - 20 \end{bmatrix} U(s) \tag{29}$$

The estimated transfer functions of Equation (29) are identical to the actual transfer functions of Equation (28). Response comparison for $Y_1(s)/U_1(s)$ transfer function estimation is shown from the estimated transfer function estimation is shown in Figure 1. The next step is to develop time domain equations from the estimated transfer functions. Phase variable, time-domain equations (14) are first determined with respect to the first input. The second input matrix is determined as solutions of three sets of linear equations (20) and also as the least squares solution of Equation (26). Transformation to F and G form is possible since all the states are measured directly. The F matrix and the G matrix obtained as solutions from both the linear sets of equations and the least squares problem are given by Equation (30). The time-domain model developed from the estimated transfer functions is identical to the actual model of Equation (29).

$$F = \begin{bmatrix} 9.22x10^{-6} & 3.53x10^{-6} & .999 \\ -10 & -3 & 8.72x10^{-6} \\ -9.83x10^{-6} & 2 & -2.999 \end{bmatrix} \qquad G = \overset{\text{Set \#1}}{\begin{bmatrix} 3.51x10^{-6} & 1 \\ 10 & -4x10^{-6} \\ 5.4x10^{-6} & -1.2x10^{-5} \end{bmatrix}}$$

$$\overset{\text{Set \#2}}{\begin{bmatrix} -3.51x10^{-6} & .999 \\ 10 & -6.32x10^{-6} \\ 5.4x10^{-6} & -4.9x10^{-6} \end{bmatrix}} \overset{\text{Set \#3}}{\begin{bmatrix} -3.51x10^{-6} & .999 \\ 10 & 2.34x10^{-5} \\ 5.4x10^{-6} & 3.51x10^{-6} \end{bmatrix}} \overset{\text{Least Squares Sol.}}{\begin{bmatrix} -3.51x10^{-6} & .999 \\ 10 & -1.1x10^{-5} \\ 5.4x10^{-6} & 7.1x10^{-7} \end{bmatrix}} \tag{30}$$

5. CONCLUDING REMARKS

Modeling procedures have been presented to develop simplified time-domain equations of motion of a SCR aircraft including the flexible and unsteady aerodynamic effects. The modeling procedures involved two steps: estimating the transfer functions from frequency response data and converting the transfer functions to state-space form. The second task of the study involves development of time-domain equations from aircraft response data for handling qualities evaluation.

REFERENCES

1. Chalk, C. R.: "Task Plan Development of Modeling Procedures and Potential Ground/Flight Simulator Experiments for Evaluation of Handling Qualities of Flexible SCR Aircraft," TIFS Memo No. 947, 8 December 1980.

2. Wykes, J. H., Byer, T. R., MacMiller, C. J., and Greek, D. C.: "Analyses and Tests of the B-1 Structural Mode Control System," NASA CR-144887, January 1980.

3. Eulrich, B. J. and Rynaski, E. G.: "Identification of Flexible Aircraft from Flight Data," presented at the AIAA 7th Atmospheric Flight Mechanics Conference, Danvers, Massachusetts, August 11-13, 1980.

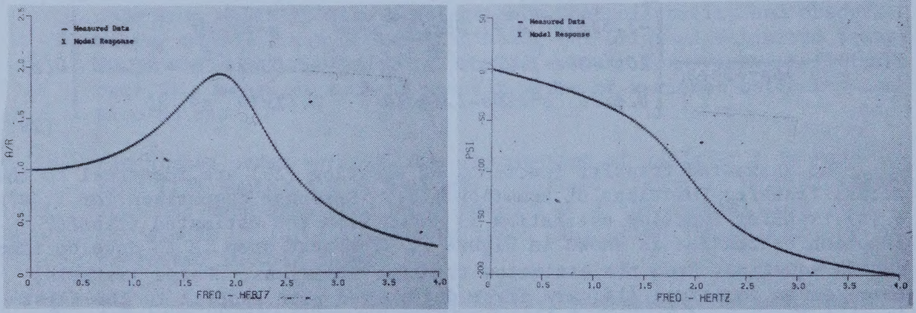


Figure 1. RESPONSE COMPARISON OF AMPLITUDE AND PHASE

APPLICATION OF MAXIMUM LIKELIHOOD ESTIMATION TO THE IDENTIFICATION OF THE STABILITY DERIVATIVES OF A WIDE BODY TRANSPORT AIRCRAFT

K. M. SOBEL and E. Y. SHAPIRO
Lockheed-California Company
Burbank, California 91520

ABSTRACT

Maximum likelihood estimation is applied to data obtained from a simulation of the lateral dynamics of an L-1011 aircraft at a cruise flight condition. The model includes a stability augmentation system (SAS), actuator dynamics, sensor noise, and random gust effects. The relative importance of the various stability derivatives and the results of the identification procedure are discussed.

I. INTRODUCTION

In order to design a flight control system for an aircraft, it is important to have a model which describes its dynamic behavior with sufficient accuracy. A typical approach is to obtain linear time invariant (LTI) models at selected points on the flight envelope. Once the form of the model is chosen; it remains to determine the parameters is this model. Such a task is known as parameter identification. This paper presents an application of maximum likelihood estimation to data obtained from a simulation of the lateral dynamics of the L-1011 aircraft at a cruise flight condition.

Many researchers have considered maximum likelihood estimation due to its well known asymptotic properties of producing unbiased, consistent, and efficient estimates. Gupta and Mehra [1] have addressed computational aspects of ML estimation, Mehra and Tyler [2] have applied ML estimation to VTOL aircraft and lifting bodies, and Best [3] has reviewed the theoretical and computational aspects of ML estimation. Maine and Iliff [4,5] have applied ML estimation to the identification of aircraft stability derivatives. However, due to the interactions between the parameters it may be difficult to obtain good estimates. Thus, this paper introduces the multi-sweep method in which several stability derivatives are identified during each sweep of the flight data.

This paper is organized as follows. Section 2 presents the aircraft model including a description of the stability augmentation system (SAS). The relative importance of the various stability derivatives is discussed. Section 3 presents the multi-sweep method; then the states of the aircraft with the true parameters are shown overplotted with the states of the aircraft with the estimated parameters. Section 4 contains conclusions and areas for future research.

2. AIRCRAFT MODEL

The model for simulation should be chosen to reflect as accurately as possible realistic flight conditions. Thus, the model, which represents the lateral dynamics of an L-1011 aircraft, includes a stability augmentation system, actuator dynamics, sensor noise and gust effects. The state space form of the model is:

$$\dot{\underline{x}}(t) = A\underline{x}(t) + B\,\underline{u}(t) + E\,\underline{w}(t) \tag{2-1}$$

$$\underline{y}(t) = C\underline{x}(t) + \underline{v}(t) \tag{2-2}$$

where

$\underline{x}(t)$ is the n x 1 state vector
$\underline{u}(t)$ is the m x 1 imput vector
$\underline{w}(t)$ is the q x 1 state disturbance vector
$\underline{y}(t)$ is the r x 1 output vector
$\underline{v}(t)$ is the r x 1 measurement noise vector

2.1 STABILITY DERIVATIVES

The open loop lateral rigid body dynamics of the L-1011 aircraft at 33,000 ft. with an airspeed of Mach. 85, may be represented by:

$$\begin{bmatrix} \dot{r} \\ \dot{\phi} \\ \dot{p} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} N_r & 0 & N_p & N_\beta \\ 0 & 0 & 1 & 0 \\ L_r & 0 & L_p & L_\beta \\ Y_r & g/u_o & Y_p & Y_\beta \end{bmatrix} \begin{bmatrix} r \\ \phi \\ p \\ \beta \end{bmatrix} + \begin{bmatrix} N_{\delta_r} & 0 \\ 0 & 0 \\ L_{\delta_r} & L_{\delta_a} \\ Y_{\delta_r} & 0 \end{bmatrix} \begin{bmatrix} \delta_r \\ \delta_a \end{bmatrix} \tag{2-3}$$

where

r = yaw rate
ϕ = bank angle
p = roll rate
β = sideslip angle
δ_r = rudder deflection
δ_a = aileron deflection

The relative importance of the stability derivatives in lateral dynamics computations are shown below [6]:

Important: N_β, N_r, L_β, L_p, L_{δ_a}

Less Important: Y_β, N_p, N_{δ_r}, L_r

Least Important: Y_r, Y_p, Y_{δ_r}, L_{δ_r}

2.2 ACTUATORS, SENSOR, AND GUST EFFECTS

The rudder and aileron actuators will be modelled by the following transfer functions:

$$\frac{\delta_r}{\delta_{rc}} = \frac{20}{s + 20} \tag{2-4}$$

and

$$\frac{\delta_a}{\delta_{ac}} = \frac{25}{s + 25} \tag{2-5}$$

where

δ_{rc} = rudder command

δ_{ac} = aileron command

Therefore, the pilot commands and stability augmentation signals will be inputs to the systems described by (2-4) and (2-5).

The aircraft measurements are chosen to be r, ϕ, p, β, δ_a, and δ_r. Sensor noise parameters are shown in Table I [7].

The wind state $\Delta\beta(t)$ influences the lateral dynamics in the same manner as the sideslip angle [8]. Thus, in the lateral state equations the wind state $\Delta\beta(t)$ enters the equations as follows:

$$\begin{aligned} \dot{r}(t) &= \ldots + N_\beta \, \Delta\beta(t) \\ \dot{p}(t) &= \ldots + L_\beta \, \Delta\beta(t) \\ \dot{\beta}(t) &= \ldots + Y_\beta \, \Delta\beta(t) \end{aligned} \tag{2-6}$$

where N_β, L_β, and Y_β can be found from the open loop dynamics given by (2-3). The wind state $\Delta\beta(t)$ is chosen to be white gaussian noise with a standard deviation σ = 1 ft/sec (which is then normalized to the aircraft's velocity).

2.3 STABILITY AUGMENTATION SYSTEM

The L-1011 lateral stability augmentation system (SAS) is described by the following transfer function:

$$\frac{(\delta_{r_c})\,\text{SAS}}{r} = \frac{15\,s}{s^2 + 5.25s + 1.25} \tag{2-7}$$

Alternately, the SAS may be represented by the following state space model:

$$\begin{bmatrix} \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1.25 & -5.25 \end{bmatrix} \begin{bmatrix} x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 \\ 3.75 \end{bmatrix} r \tag{2-8}$$

$$(\delta_{r_c})\text{SAS} = (0 \quad 4) \begin{bmatrix} x_5 \\ x_6 \end{bmatrix} \tag{2-9}$$

where x_5 and x_6 are the SAS states. The SAS is basically a yaw damper with a washout filter to allow proper turn coordination.

2.4 COMPOSITE MODEL

Combining the results of sections 2.1 - 2.3 the model used for simulation is obtained. The composite model is shown below:

$$\begin{bmatrix} \dot{r} \\ \dot{\phi} \\ \dot{p} \\ \dot{\beta} \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{\delta}_r \\ \dot{\delta}_a \end{bmatrix} = \begin{bmatrix} -.1544 & 0 & -.0042 & 1.54 & 0 & 0 & -.744 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ .2489 & 0 & -1.001 & -5.209 & 0 & 0 & .3377 & -1.119 \\ -.997 & .03858 & -.000295 & -.1171 & 0 & 0 & .0197 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 3.75 & 0 & 0 & 0 & -1.25 & -5.25 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 80 & -20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -25 \end{bmatrix} \begin{bmatrix} r \\ \phi \\ p \\ \beta \\ x_5 \\ x_6 \\ \delta_r \\ \delta_a \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 20 & 0 \\ 0 & 25 \end{bmatrix} \begin{bmatrix} \delta_{rc} \\ \delta_{ac} \end{bmatrix}$$

$$+ \begin{bmatrix} 1.54 \\ 0 \\ -5.209 \\ -.1171 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \Delta\beta \qquad (2\text{-}10)$$

$$\underline{y}(t) = \begin{bmatrix} \dot{r} \\ \phi \\ p \\ \beta \\ \delta_r \\ \delta_a \end{bmatrix} + \underline{v}(t) \qquad (2\text{-}11)$$

where $\underline{v}(t)$ is a white gaussian sensor noise vector.

Both inputs are chosen to be a sequence of trapezoidal pulses with maximum amplitude of approximately 3 degrees. Such a command signal is reasonably within the pilot's ability. The input signal is shown in Figure 1.

3. IDENTIFICATION OF THE STABILITY DERIVATIVES

Before initiating the identification procedure it is necessary to determine which parameters do not need to be identified. Recall that Y_r, Y_p, and $Y_{\delta r}$ are the least important of the stability derivatives. Also, Y_r is reasonably known a priori. Thus, Y_r, Y_p, and $Y_{\delta r}$ are fixed at the following values:

$$Y_r - -1, \; Y_p = 0, \; Y_{\delta r} = 0$$

The initial estimates for the remaining stability derivatives are chosen to reflect the parameter uncertainties described in [9].

The first step consists of sweeping through the data which was generated by rudder command only. The stability derivatives to be identified are N_β, L_p, L_β, Y_β, $N_{\delta r}$, N_r and $L_{\delta r}$. Observe that N_β, L_β, and Y_β appear in both the A matrix and E matrix of eqn. (2-10). The initial estimates, final estimates, and standard deviations are shown in Table II.

The second step consists of repeating the first step with the elements of the E matrix frozen at their final estimates from the first step. The parameters to be identified in the A matrix of eqn. (2-10) are initialized at their final values from the first step except for $L_{\delta r}$. The ML estimate for $L_{\delta r}$ from the first step has a large standard deviation. Thus, it is again initialized at its a priori value. The initial estimates, final estimates, and standard deviations are shown in Table III.

The final step consists of sweeping through the data which was generated by aileron command only. The stability derivatives to be identified are N_p, L_r, and L_{δ_a}. The other unknown parameters in both the A matrix and the E matrix are fixed at the ML estimates from the second sweep except for $L_{\delta r}$. $L_{\delta r}$ is fixed at its a priori value because its ML estimate has a large standard deviation. Also the confidence in the a priori value of $L_{\delta r}$ is approximately ± 25% [9]. The initial estimates, final estimates and standared deviations are shown in Table IV. Observe that the standard deviations for N_p and L_r are large. This is not entirely unexpected because aerodynamicists have difficulty obtaining these quantities via conventional methods. Also, these parameters are not among those which are of primary importance in lateral dynamics calculations.

Figure 2 shows the states of the aircraft using the true parameters overplotted with the states of the aircraft using the ML estimates shown in Table IV. For this comparison the input signal shown in Figure 1 commands both rudder and aileron. The roll rates, yaw rates, bank angles, and sideslip angles are sufficiently close. Thus, the stability derivatives have been estimated with sufficient accuracy.

4. CONCLUSIONS

This paper has presented an application of maximum likelihood estimation to the rigid body lateral dynamics of an L-1011 aircraft at a cruise flight condition. The multi-sweep method, in which several stability derivatives are identified during each sweep of the data, was introduced. This method resulted in the identification of the required parameters with sufficient accuracy. Future work will include using the Von Karman model for gust effects, obtaining data by using a nonlinear simulation, and using actual flight test data.

REFERENCES

1. Gupta, N. K. and Mehra, R. K. "Computational Aspects of Maximum Likelihood Estimation and Reduction in Sensitivity Function Generation," IEEE Transactions on Automatic Control, Vol. AC-19, No. 6, December 1974.

2. Mehra, R. K. and Tyler, J. S., "Case Studies in Aircraft Parameter Identification," Proceedings of the 3rd IFAC Symposium, The Hague/Delft The Netherlands, June 1973.

3. Best, J. R. "Parameter Identification of Linear Systems Via The Maximum Likelihood Method," M. S. Thesis, The University of Connecticut, 1978.

4. Iliff, K. W. and Maine, R. E., "Practical Aspects of Using a Maximum Likelihood Estimation Method to Extract Stability and Control Derivatives from Flight Data," NASA TDN-8209, 1976.

5. Iliff, K. W. and Maine, R. E., "Further Observations on Maximum Likelihood Estimates of Stability and Control Characteristics Obtained From Flight Data," AIAA Paper 77-1133, AIAA Atmospheric Flight Mechanics Conference, Hollywood, Fla. August 1977.

6. Dynamics of the Airframe, Bureau of Aeronautics, Navy Department, September 1952.

7. Elliott, J. R., "NASA's Advanced Control Law Program For the F-8 Digital Fly-by-Wire Aircraft," IEEE Transactions on Automatic Control, Vol. AC-22, No. 5, October 1977.

8. Athans, M. et al, "The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control Method - Part I: Equilibrium Flight," IEEE Transactions on Automatic Control, Vol. AC-22, No. 5, October 1977.

9. Shapiro, E.Y., "Software Techniques for Sensor Redundancy Management of Flight Control Systems," Journal of Aircraft, Vol. 14, No. 7, July 1977.

TABLE I

SENSOR NOISE PARAMETERS

SENSOR	STANDARD DEVIATION
Yaw Rate Gyro	0.15 deg/sec
Roll Rate Gyro	0.75 deg/sec
Bank Angle	0.05 deg
Sideslip Angle	0.10 deg
Servo Position	0.04 deg

TABLE II

RESULTS OF FIRST SWEEP

	INITIAL ESTIMATE	FINAL ESTIMATE	STANDARD DEVIATION
N_r	-0.139	-0.146	0.0568
N_p	0.0	Fixed	- - - -
N_β	3.0	1.42	0.0771
$N_{\delta r}$	-0.558	-0.678	0.0722
L_r	0.0	Fixed	- - - -
L_p	0.0	-1.14	0.0792
L_β	-10.4	-5.42	0.402
$L_{\delta r}$	0.253	0.156	0.263
$L_{\delta a}$	-0.95	Fixed	- - - -
Y_r	-1.0	Fixed	- - - -
Y_p	0.0	Fixed	- - - -
Y_β	-0.25	-0.0903	0.00588
$Y_{\delta r}$	0.0	Fixed	- - - -

TABLE III

RESULTS OF SECOND SWEEP

* The parameters in the E matrix are fixed.

	INITIAL ESTIMATE	FINAL ESTIMATE	STANDARD DEVIATION
N_r	-0.146	-0.167	0.0573
N_p	0.0	Fixed	- - - -
N_β^*	1.42	1.54	0.0832
$N_{\delta r}$	-0.678	-0.732	0.0754
L_r	0.0	Fixed	- - - -
L_p	-1.14	-1.09	0.0802
L_β^*	-5.42	-5.24	0.417
$L_{\delta r}$	0.253	0.127	0.268
$L_{\delta a}$	-0.95	Fixed	- - - -
Y_r	-1.0	Fixed	- - - -
Y_p	0.0	Fixed	- - - -
Y_β^*	-0.0903	-0.0857	0.00575
$Y_{\delta r}$	0.0	Fixed	- - - -

TABLE IV

RESULTS OF THIRD SWEEP

	INITIAL ESTIMATE	FINAL ESTIMATE	STANDARD DEVIATION
N_r	-0.167	Fixed	- - - -
N_p	0.0	-0.00234	0.0184
N_β	1.54	Fixed	- - - -
$N_{\delta r}$	-0.732	Fixed	- - - -
L_r	0.0	0.275	1.50
L_p	-1.09	Fixed	- - - -
L_β	-5.24	Fixed	- - - -
$L_{\delta r}$	0.253	Fixed	- - - -
$L_{\delta a}$	-0.95	-1.17	0.0606
Y_r	-1.0	Fixed	- - - -
Y_p	0.0	Fixed	- - - -
Y_β	-0.0903	Fixed	- - - -
$Y_{\delta r}$	0.0	Fixed	- - - -

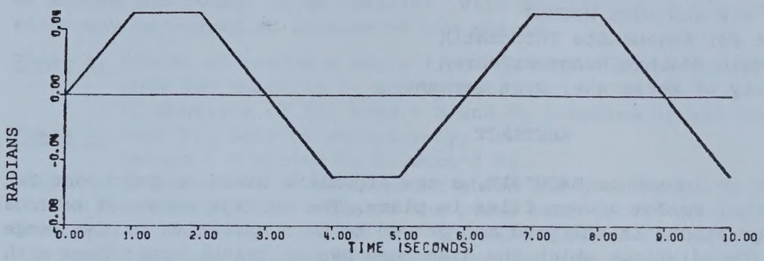


FIGURE 1
PILOT COMMAND

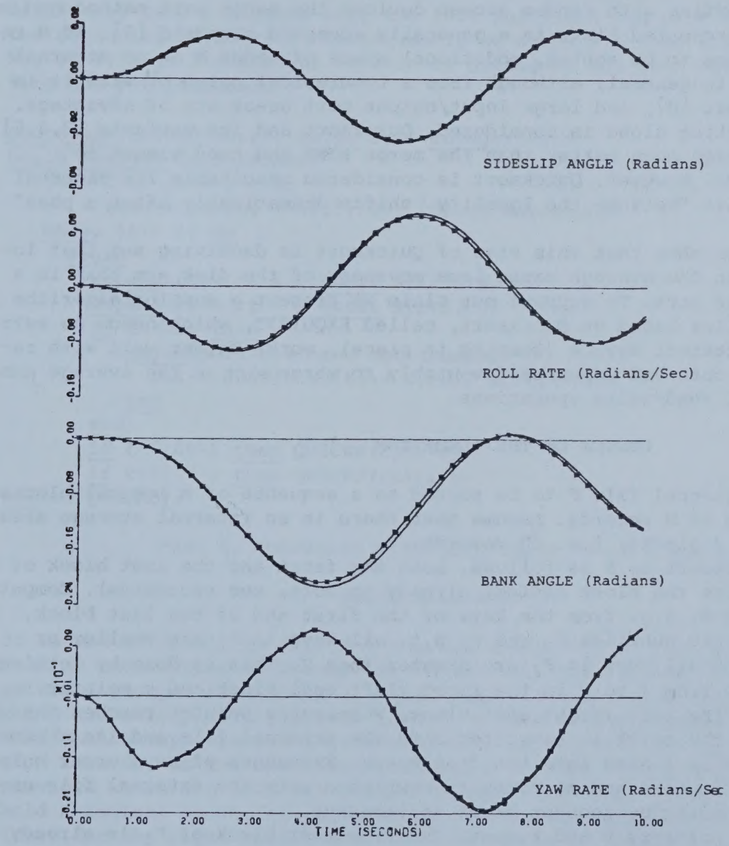


FIGURE 2
STATE OVERPLOTS

EXQUISIT: APPLYING QUICKSORT TO EXTERNAL FILES

HANS W. SIX and LUTZ WEGNER
Institut für Angewandte Informatik
und Formale Beschreibungsverfahren
University of Karlsruhe, West Germany

ABSTRACT

In this paper we introduce EXQUISIT, a new algorithm based on Quicksort for sorting external random access files in place. The average number of external read/write operations is analysed and proven to be competetive to the merge sort. As for the distance which the disk head has to travel, test runs with EXQUISIT show a substantial improvement over the merge sort, if the latter does not use an interleaving technique.

EXTERNAL SORTING REVISITED

For external sorting with random access devices the merge sort method applied to internally presorted lists is a generally accepted standard [2]. If N records of data are to be sorted, additional space of order N on an external device is used in general, although from a theoretical point of view it is not a prerequisit [8], and large input/output sort areas are of advantage. If internal sorting alone is considered, Quicksort and its variants [3,4,5] are on the average much better than the merge sort and need almost no additional space. However, Quicksort is considered unsuitable for external purposes, in part "because the locality shifts dramatically after a pass" [1, p. 342].

In this paper we show that this view of Quicksort is deceiving and that locality shifts on the average cause less movement of the disk arm than in a comparable merge sort. To support our claim we present a sorting algorithm for external files based on Quicksort, called EXQUISIT, which needs no extra space on the external device (sorting in place), works rather well with relatively small core and compares favourably to merge sort in the average number of external read/write operations.

DESIGN OF THE ALGORITHM

Consider the external file F to be sorted as a sequence of M logical blocks each consisting of N records. Assume that there is an internal storage area which may hold 2 blocks, i.e. 2N records.

Then apply Quicksort to F as follows. Load the first and the last block of F into core (unless the block resides already in core, see recursion). Compute a pivot element H, e.g. from the keys of the first and of the last block. Then devide F into subfiles F_1 and F_2 s.t. all keys in F_1 are smaller or equal than H and all keys in F_2 are greater than H. This is done by letting a pointer ℓ run from 1 to N in the first (left end) block and a pointer r from N to 1 in the last (right end) block. Whenever a pointer reaches the block boarder, the block is rewritten onto the external file and its adjacent neighbour block is loaded into the free space. Exchanges of keys occur only between blocks in core and no block is rewritten onto the external file unless its overloading by another block is immanent.

Thus, when the pointers ℓ and r meet, the rightmost block of F_1 is already in core and the recursion for F_1 may start with loading the first block of F_1. Moreover, when the recursion for F_2 starts, in most cases the first block of F_2 will again be available at no extra cost. Obviously recursion always ends with subfiles that lie entirely in one block.

ADAPTING QUICKSORT TO EXTERNAL FILES

We assume the reader to be familiar with Quicksort [3]. In short this algorithm may be viewed to consist of two phases:

Phase 1: choose at random a key H from the file F to be sorted and devide F
into two subfiles F_1 and F_2 s.t.
F_1 consists of all keys $\leq$ H and F_2 consists of all keys $>$ H.
Phase 2: sort F_1, sort F_2 recursively;
return F:= sorted F_1,H, sorted F_2.

Adapting Quicksort to an environment where the data reside on external random access devices precludes some of the usual techniques. Obviously it is not desirable to exchange H at the end of phase 1 since the record belonging to key H will most likely not be in core by the time it is needed. As a result the first design measure taken is to use a so called fictitious pivot element H whose final placement may be thought to lie on the border between two adjacent keys (the last key of F_1 and the first key of F_2).
Secondly, only those keys may be used to compute the fictitious pivot element H which are in core at the time the choice is made. This excludes the middle key from F because only the left and right end of F are on hand at the beginning of phase 1. Consequently we compute H from the first and last key of F. Fig. 1 shows our version of Quicksort but with external file handling not yet included.

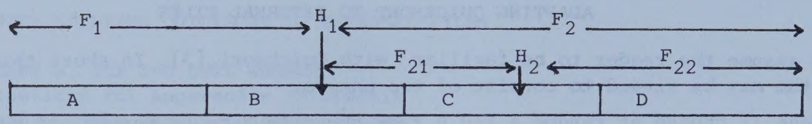
```
procedure QUICKS (ℓ,r); value ℓ,r; integer ℓ,r;
begin integer ℓt,rt,H,aux;
  ℓt:= ℓ-1; rt:= r + 1;
  H := entier ((s[ℓ] + s[r])/2); {pivot selection}
  while ℓt < rt do
    begin
      repeat ℓt:= ℓt + 1 until s[ℓt] ≥ H;
      repeat rt:= rt - 1 until s[rt] ≤ H;
      if ℓt < rt then
        begin{pointers have not crossed}
          aux:= s[ℓt]; s[ℓt]:= s[rt]; s[rt]:= aux
        end
      end;
      if ℓ < ℓt-1 then QUICKS(ℓ,ℓt-1);
      if rt+1 < r then QUICKS(rt+1,r)
end{of QUICKS};
```

Fig. 1: Quicksort with fictitious pivot element

BLOCK HANDLING

It is not difficult to extend the procedure QUICKS of Fig. 1 s.t. it handles reading and writing of blocks. We assume that the records of the file F are as usual grouped into physical blocks and data are transfered in logical blocks which are multiples of physical blocks. A read or write request from EXQUISIT will then always concern a logical block i.e. it will result in the transfer of N records. As for the internal space we assume that a sorting area S is provided which can hold two logical blocks. Note that the left half of S may hold a logical block from the "left end" of F or from the "right end" of F (same for the right half). Thus the management of blocks in based on a virtual addressing scheme which also handles the case where one and the same block in core is seen to stem from the left and from the right end of F.

Figure 2 shows a possible history of core occupation for a file of 4 blocks.



left half of storage area	right half of storage area	comment
A	D	initial load
B	↓	ℓ crossed border from A to B
↓	C	r crossed border from D to C
	A	ℓ and r met in B, H_1 is pivot location,
		one load for F_1
↓	D	initial load for F_2
C	↓	ℓ crossed border from B to C
↓	B	one load for F_{21}

Fig. 2: A history of core occupation for a file of 4 blocks

ANALYTICAL RESULTS

(For detailed arguments of this section the reader is conferred to [9]). The analysis of EXQUISIT centers around the number of logical block reads. As it turns out the analysis is somewhat complicated because probabilities for the position of the files and subfiles with respect to block borders enter the calculation. However, we are able to prove the following: Given a file F which stretches over k blocks and which has been split according to the fictitious pivot element H into subfiles F_1 and F_2, it is possible to split F_1 and after completely sorting F_1 to split F_2 with k-1 block reads altogether regardless of the position of H in F.

Using this strangely twisted starting point (F already split) and entering the probabilities we can formulate a recurrence relation which yields about $2k \cdot \ln k + 3k$ as an upper bound for the average number of block reads to sort a file consisting of k blocks.

For the course of this discussion let x be the size of the file F, i.e. the number of records (keys) in F and let L(F) denote the position of the first record of F in the first block. As usual for this kind of analysis (see e.g. [4]) we shall make use of the following assumptions.

Assumptions

(i) The file F is in random order and the keys to be sorted are distinct.

(ii) L(F) is uniformly distributed over $\{1,\dots,N\}$, i.e. all starting positions of F are equally likely.

(iii) Let file F of size x be split according to a (fictitious) pivot element into two subfiles F_1 of size s and F_2 of size x-s ($1 \leq s \leq x-1$). Then any value of s occurs with probability $\frac{1}{x-1}$.

The recurrence relation mentioned above averages over all starting positions of F, all sizes of the subfiles F_1 and F_2 after partitioning and all starting positions of the subfiles. As it turns out, the main problem is that we may not assume that, at the beginning of a partitioning phase, one of the blocks of the subfile under consideration is always in core. As the following Lemma shows, this anomality occurs when the split happens to fall onto a block border on the previous level of recursion. Fortunately, to start with a file already split does the trick, as shown in the following Lemma.

Lemma: Let F be a file which stretches over $k \geq 5$ blocks. Let F be split into F_1 and F_2 both stretching over more than two blocks. Then for all $s \in \{1,\dots,x-1\}$ it takes k-1 block reads to split F_1 and, after completely sorting F_1, to split F_2.

Proof: We distinguish two cases:
a) the split happens to fall within block i ($1 \leq i \leq k$)
b) the split happens to fall onto a block border.
The situation is depicted in Figure 3 for k=6.

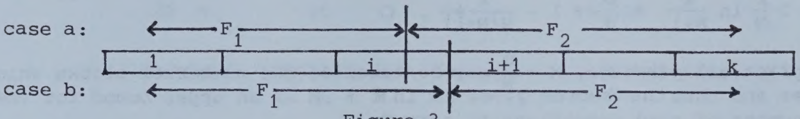


Figure 3

Since in both cases after the split of F block i is in core it always takes i-1 block loads to split F_1. According to the preorder strategy block i is in core again when F_2 is about to be sorted. Hence we have k-i block reads to split F_2 in case a and b and therefore it takes k-1 load operations to split F_1 and F_2. □

If F_1 or F_2 is shorter than 3 blocks, splitting F_1 and F_2 may require less than k-1 read operations. Thus k-1 remains as upper bound for the partitioning phase up to the point, where the file has size N+1. Clearly F now stretches over at most two blocks and no further block reads are needed to sort F (boundary condition).
Before entering the above result into the recurrence relation, care has to be taken in showing that the assumptions (i) - (iii) above hold again for the subfiles F_1 and F_2 after partitioning.

Lemma: Let $k_{L(F)}$ denote the number of blocks occupied by F with starting position L(F). The recurrence relation

$$g(x) = \frac{1}{N} \sum_{1 \leq L(F) \leq N} (k_{L(F)} - 1) + \frac{1}{x-1} \sum_{1 \leq s \leq x-1} (g(s) + g(x-s)) \quad \text{for } x > N+1$$

and $g(x) = 0$ for $x \leq N+1$ gives an upper bound for the average number of block reads to sort F of size x when F has already been split. □

The average number of blocks occupied by F is given by the term $\frac{1}{N} \sum_{1 \leq L(F) \leq N} k_{L(F)}$ whose evaluation leads to $\frac{x-1}{N} + 1$.
We are now in a position to give an upper bound on the expected number of block reads EBR(x) to sort a file F of size x under the assumptions (i)-(iii). EBR(x) is computed from the upper bound g(x) above and the initial effort to split the original file F.

Theorem: $EBR(x) \leq 2\frac{x}{N} \ln \frac{x}{N+1} + 3\frac{x}{N} + 1 - \frac{x}{N(N+1)}$.

Proof: Since $\frac{1}{N} \sum_{1 \leq L(F) \leq N} k_{L(F)} = \frac{x-1}{N} + 1$ we arrive at

$$g(x) = \frac{x-1}{N} + \frac{1}{x-1} \sum_{1 \leq s \leq x-1} (g(s) + g(x-s)) \quad \text{for } x > N+1$$

$$\text{and } g(x) = 0 \quad \text{for } x \leq N+1$$

The recurrence relation g(x) can be solved using the familiar techniques of "eliminating the sum" and "telescoping" (cf. [6]).

$$g(x) = \frac{x}{N}\left[2 \cdot \sum_{i=N+2}^{x} \frac{1}{i} - \sum_{i=N+2}^{x-1} \frac{1}{i(i+1)}\right] \leq 2\frac{x}{N}\left[\ln \frac{x}{N+1} + 1 + \frac{1}{2x} - \frac{1}{2(N+1)}\right].$$

Adding the effort $\frac{x-1}{N} + 1$ for the initial split of F we have

$$EBR(x) \leq 2\cdot\frac{x}{N}\left[\ln\frac{x}{N+1} + 1 + \frac{1}{2x} - \frac{1}{2(N+1)}\right] + \frac{x-1}{N} + 1$$

$$= 2\cdot\frac{x}{N}\cdot\ln\frac{x}{N+1} + \frac{3x}{N} + 1 - \frac{x}{N(N+1)}\ . \quad \square$$

For all practical purposes, $M = \frac{x}{N}$ may be taken as the number of blocks which F occupies and thus the Theorem gives $2M \ln M + 3M$ as an upper bound for the average number of read operations to sort F.

A few words have still to be said about the worst case behavior. Since a file of M blocks has a size $x \approx M\cdot N$, note that when each partition phase splits off one element only, a worst case of order $M^2\cdot N$ results.

Hence, a safe strategy for the choice of the pivot element is even more important for the external than for the internal Quicksort. Taking keys from the first and last block of F is a good choice. The best solution in our opinion, however, is to compute the fictitious pivot element from the minimal and maximal key in the (sub-)file to be sorted. This sort procedure is sometimes called Binary Sort (cf. [7]). In order to force EXQUISIT with the $(\min(F) + \max(F))/2$ pivot selection into a $O(M^2\cdot N)$-type degeneration, the values of the keys have to increase exponentially which quickly finds its limit in the key size.

More on this strategy may be found in [9] where we also present a detailed discussion of the experimental findings. In short they confirme that $2M \ln M + 3M$ is not a sharp bound because of the influence of subfiles of length ≤ 5 blocks and because of the improved pivot selection technique. As the table below shows, $1.5M \ln M$ seems to be a good estimate on the number of block reads which may be expected on the average.

EXPERIMENTAL RESULTS

We directly compare EXQUISIT with a suitable version of a p-way merge sort. Since EXQUISIT's implementation is straightforward, a few words have to be said about the merge sort only.

The p-way merge sort starts as usual with forming presorted strings of length equal to the core size. For each test run the optimal merge order p_{opt} is computed as the value of p which minimizes the total number of external reads (and writes). An interleaving technique (cf. [5]) is not implemented.

It is important to mention that in contrast to EXQUISIT the (logical) blocksize used in the merge sort is not half the core size. Since during a p-way merge the core is devided into p input buffer and one output buffer, the (logical) blocksize of the merge sort is set to core size/(p+1). Thus EXQUISIT transfers data in larger portions - except during the presorting phase of the merge sort where each filling of the core is counted as one block read.

Three counts are implemented in both sort programs:

1. NBR : no. of block reads
2. NBW : no. of block writes
3. DHT : distance of read/write head travelled (measured in records).

The file to be sorted consists of 2400 keys which are the integers 1,2,...,2400 in a random permutation and the core size varies from 0,5% to 20% of the file size. We present two tables of results for EXQUISIT: Table 1 gives the figures when the "(min(F) + max(F))/2 strategy" is used, Table 2 those obtained by a "(first key + last key)/2 strategy". Table 3 shows the results for the merge sort.

core	NBR	NBW	DHT
0.5%	3643	3540	779.490
1 %	1657	1642	444.336
5 %	215	210	107.820
20 %	35	34	35.520

Table 1: EXQUISIT with "(min(F) + max(F))/2 strategy"

core	NBR	NBW	DHT
0.5%	4262	3726	786.960
1 %	1953	1832	440.352
5 %	300	298	132.120
20 %	48	48	46.560

Table 2: EXQUISIT with "(first + last key)/2 strategy"

core	NBR (=NBW)	DHT	P_{opt}
0.5%	4200	15.911.952	3
1 %	1900	6.707.080	5
5 %	260	903.960	3
20 %	35	113.600	5

Table 3: Merge sort

The above figures indicate that EXQUISIT is a true competitor to the merge sort beating the latter roughly 1 to 10 in the distance the head has to travel when the merge sort does not use an interleaving technique. Moreover, in our simulation environment the "(min(F) + max(F))/2 strategy" causes about 15% less external accesses than the "(first key + last key)/2" strategy.

CONCLUSION

The analysis of the algorithm and the experimental results clearly indicate that EXQUISIT is an interesting alternative to the merge sort for sorting a random access file. Among the possible improvements, the following seems most promissing. If the logical block size is much larger than the physical block size, then whenever a subfile stretches some way into a logical block but not through the entire block, only those physical blocks are loaded which are actually needed. Note that all the necessary information for doing so is available at the time a load request is issued.

What cannot be repaired so easily are two inherent weaknesses of Quicksort: the sort is not stable and failure to complete the sort may result in a loss of data. A new linked list approach remedies these defects (report in preparation).

REFERENCES

[1] N.A. Black: Optimum Merging from Mass Storage, Comm. ACM 13 (1970), 745-749

[2] B.S. Brawn, F.G. Gustavson and E. Mankin : Sorting in a Paging Environment, Comm. ACM 13 (1970), 483-494

[3] C.A.R. Hoare: Quicksort, Comput. J. 5 (1962), 10-15

[4] Donald E. Knuth: The Art of Computer Programming, Vol. 3, Addison-Wesley, Reading, Mass., 1973

[5] Harold Lorin: Sorting and Sort Systems, Addison-Wesley, Reading, Mass. 1975

[6] George Lueker: Some Techniques for Solving Recurrences, Comp. Surv. 12 (1980), 419-436

[7] Hermann Maurer: Data structures and Programming, Prentice Hall, Englewood Cliffs, N.J., 1973

[8] Luis Trabb Pardo: Stable Sorting and Merging with Optimal Space and Time Bounds, Siam J. Comput., Vol. 6, No. 2 (1977), 351-372

[9] H.W. Six and L. Wegner: Sorting a Random Access File in Situ, Report 105 (July 1981), Institut für Angewandte Informatik und Formale Beschreibungsverfahren, Universität Karlsruhe.

TWO VERY FAST TRIE DATA STRUCTURES (EXTENDED ABSTRACT)

DAN E. WILLARD
Bell Laboratories, Holmdel, NJ 07733

ABSTRACT

Let S denote a set of N records whose keys are distinct nonnegative integers, less than some initially specified bound M. This paper introduces a new data structure, called the q-fast trie, which uses space $\Theta(N)$ and time $\Theta(\sqrt{\log M})$ on a Random Access Machine for insertions, deletions, and all the retrieval operations commonly associated with binary trees. We will also define a simpler but less efficient structure, called the p-fast trie.

INTRODUCTION

Below is a list of the names and descriptions of the retrieval operations, considered in this paper:

i) MEMBER(K): Determine whether or not key K belongs to set S;
ii) SUCCESSOR(K): Find the least element in set S with key value greater than K;
iii) PREDECESSOR(K): Find the greatest number in set S with key value less than K;
iv) SUBSET(K_1,K_2): Find (and then produce) the list of the elements from set S whose key values lie between K_1 and K_2.

For convenience, we will say that a data structure has a worst-case overall-retrieval complexity $\Theta[f(M)]$ iff retrieval operations i through iii have a worst-case runtime $\Theta[f(m)]$ and the worst-case runtime of SUBSET queries is proportional to f(m) plus the size of the retrieved subset.

Van Emde Boas, Kaas, and Zijlstra [9] have presented a data structure called a stratified tree that has worst-case overall-retrieval complexity $\Theta(\log \log M)$ and uses $\Theta(M \log \log M)$ memory. Their work motivated much of our research effort. The need for more space-efficient data structures was indicated in [7]. Van Emde Boas developed a modified tree data structure that uses $\Theta(M)$ space in [8]. Also, Knuth [5] offered a very clear summary of [7,8,9].

The full length version of this paper [10] proves that even in a static environment where one conserves memory, the stratified trees from [5,7,8,9] use at least expected memory $\Omega(\sqrt{NM})$ when there exists an integer i such that $N = \Theta(M^{1-2^{-i}})$, and that the worst-case space always respects the lower bound $\Omega(N^{3/4}M^{1/4})$. Therefore, when N is much less than M, q-fast tries use significantly less space than any form of stratified tree and are preferable in a large body of applications.*

Our results are also interesting because log N is a well-known theoretical lower bound on the worst-case runtime for a random access machine to determine if a particular real number is among an ordered list of N real numbers [4]. We show that faster runtimes in memory space $\Theta(N)$ are possible if the keys are integers and if $\sqrt{\log M} < \log N$. This result is significant because the latter inequality holds for almost all values of M and N, including most cases where M is much greater than N. For instance, if $M = 2^{100}$ and $N = 2^{20}$ then $\sqrt{\log M} = 10$ and $\log N = 20$.

THE ORGANIZATION OF THE PAPER AND ITS GENERAL PERSPECTIVE

We have divided the discussion in the full-length version of this paper [10] into three major parts. The first section introduces a simplified data structure, called the p-fast trie, that has overall worst-case retrieval complexity $\Theta(\sqrt{\log M})$ and uses space $\Theta\left(N\sqrt{\log M}\, 2^{\sqrt{\log M}}\right)$. The second section shows that insertions and deletions can be performed in runtime $\Theta(\sqrt{\log M})$ in this data structure. The third section explains how a more sophisticated data structure, called the q-fast trie, reduces memory space consumption to $\Theta(N)$. This extended abstract discusses the first topic in great detail and sketches the intuition behind the third.

As is suggested by their names, the two fast trie data structures will be modified forms of tries [2,4]. A trie whose height is h and branching factor is b will be said to have size (h,b). In most of our discussion, we will treat the retrieval complexity of a trie as a function of its size (h,b). A conventional topdown retrieval algorithm will clearly perform type i through iii queries by visiting $\Theta(h)$ nodes of the trie and will perform type iv queries by visiting a quantity of nodes proportional to h plus the size of SUBSET (K_1,K_2). The retrieval time for a type i query is nevertheless very different from that of type ii through iv queries because the former needs only $\Theta(1)$ units of time to visit each node while the latter three queries require time $\Theta(b)$ to visit a node in the worst-case. A consequence of this observation is that a trie of size (h,b) will have worst-case overall retrieval complexity $\Theta(hb)$.

Our first data structure, the p-fast trie, has a worst-case overall-retrieval complexity $\Theta(h + \log b)$ and uses asymptotically the same amount of space as a conventional trie. All our other results will follow from a series of elaborate applications of the concept of a p-fast trie. It should be stressed that a complexity $\Theta(h + \log b)$ is crucial to our analysis, and satisfactory results would be impossible if p-fast tries had a complexity of $\Theta(hb)$ or even $\Theta(h \cdot \log b)$.

---

* It should be pointed out that q-fast tries never occupy more memory space than stratified trees. This is because the literature on stratified trees assumes that no two records contain the same key value, thus assuring that N is less than or equal to M.

NOTATION AND THE DEFINITION OF P-FAST TRIES

All keys, in a conventional trie of size (h,b), can clearly be regarded as nonnegative integers less than b^h. The sequence $D_1^K \; D_2^K \; \ldots \; D_h^K$ will denote the h-digit expansion of integer-key K when it is written as a number in base b. In a trie of size (h,b), this notation clearly implies that key K is stored at the D_h^K-th son of the D_{h-1}^K-th son of the ... D_1^K-st son of the trie's root (as is illustrated in Figure 1). In our discussion, the symbol $\mathrm{SONPOINTER}_v(D)$ will denote the pointer, stored inside internal node v, indicating the address of its D-th son. For instance, in Figure 1, $\mathrm{SONPOINTER}_{root}(2)$ would point to the root's leftmost existing son. Note that the root in Figure 1 does not contain any sons corresponding to the digits zero and one. By convention, we will therefore store the element NULL in $\mathrm{SONPOINTER}_{root}(0)$ and $\mathrm{SONPOINTER}_{root}(1)$.

The symbol S will always denote the set of integer keys stored in our trie. Our tries will always be structured so that their set of internal nodes is the smallest possible set. That is, we will store an internal node v in trie T if and only if it is the ancestor of some element of S. In the literature on tries, this compressed-storage representation has been used to conserve memory space. Our search algorithm is somewhat unusual because it will also need compression to guarantee its runtime efficiency, and indeed even its correctness.

We are now ready to define p-fast tries. Each internal node of these tries will contain the usual SONPOINTER fields that have been traditionally associated with tries. In addition, each internal node v will contain the following three new fields:

LOWKEY(v):	This field will be a pointer to that leaf containing the smallest key descending from v.
HIGHKEY(v):	This field will be a pointer to that leaf containing the largest key descending from v.
INNERTREE(v):	This field will consist of a binary tree of worst-case height O(log b) that represents the set of digits D such that $\mathrm{SONPOINTER}_v(D) \neq \mathrm{NULL}$.

For instance, in Figure 1, LOWKEY(root) = 20, HIGHKEY(root) = 44, and INNERTREE(root) represents the set {2,3,4}.

In addition, each leaf on a p-fast trie will be required to contain one pointer to the leaf which lies to its immediate left and another pointer

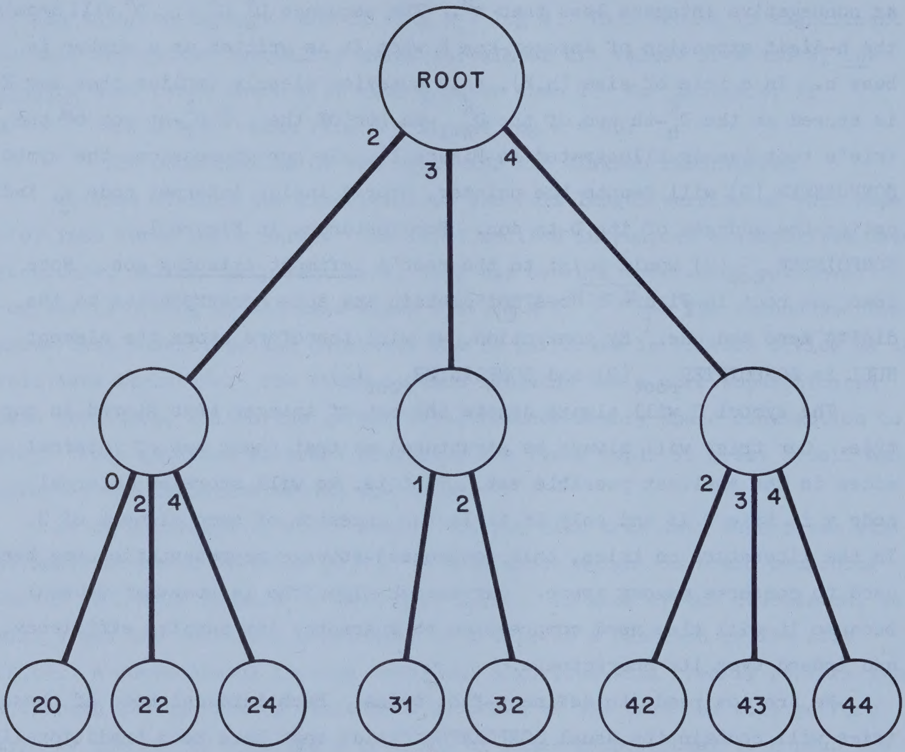

ROOT
2
3
4
0
2
4
1
2
2
3
4
20
22
24
31
32
42
43
44

FIGURE 1: AN EXAMPLE OF A TRIE WITH HEIGHT h=2 AND BRANCHING FACTOR b=5. ALL KEYS AT THE LEAF-LEVEL ARE WRITTEN AS TWO-DIGIT NUMBERS IN BASE 5.

to the leaf to its right. The sequence of keys formed in this way will be called the trie's ordered list.

COMPLEXITY ANALYSIS FOR FAST TRIES

It is useful to introduce one further definition before we begin our analysis. Define a CLOSEMATCH(K) to be a query that:

i) returns the address of the record storing key K in trie T (if the trie actually contains K);

ii) and otherwise returns the address of either PREDECESSOR(K) or SUCCESSOR(K).

In the first half of this section, we will prove that a p-fast trie of size (h,b) has a worst-case complexity $\Theta(h + \log b)$ for CLOSEMATCH requests, and in the second half we will apply this result to demonstrate that these tries support an overall worst-case retrieval complexity $\Theta(\sqrt{\log M})$.

Our algorithm for performing CLOSEMATCH(K) queries is illustrated in Figure 2. It will consist of three steps. The first step will begin at the root of the p-fast trie and will perform conventional topdown tree-walk that visits all of K's ancestors by following the obvious path indicated by the relevant SONPOINTERS. If key K is actually stored in trie T then this walk will eventually reach K, and our search algorithm will then terminate. If key K is not stored in trie T then our topdown tree-walk will eventually encounter a NULL SONPOINTER. In this latter case, our retrieval algorithm will complete its search by executing two further steps, which employ the novel characteristics of p-fast tries.

Step 2, the first of these new steps, will take advantage of the trie's special INNERTREE fields. In order to describe this procedure, we let v denote the last node that was visited by step 1 and i - 1 the depth of v. Our previous notation then implies that step 1 terminated because $\mathrm{SONPOINTER}_v(D^K_i) = \mathrm{NULL}$. Let D denote:

i) the least digit greater than D^K_i satisfying $\mathrm{SONPOINTER}_v(D) \neq \mathrm{NULL}$ (if such a digit exists);

ii) and otherwise the greatest digit less than D^K_i satisfying $\mathrm{SONPOINTER}_v(D) \neq \mathrm{NULL}$.*

A search through INNERTREE(v) can determine the value of D in clearly worst-case runtime O(log b). Step 2 will consist of such a search.

Step 3 of our retrieval algorithm for CLOSEMATCHes will use a slightly different procedure in the two cases where D is not greater than D^K_i. In the

---

* From the definition of p-fast tries, it can be easily proven that some digit must satisfy either condition i or ii. This observation is important because otherwise D could be undefined.

FIGURE 2

```
Algorithm RETRIEVE(K)
    Comment: We assume that $D_1^K$, $D_2^K$ ... $D_h^K$ denotes the expansion of key K
    as an h-digit number in base b. This algorithm will search a p-fast
    trie T of height h to find a key CLOSEMATCH(K).
    Step 0: Set v equal to the root of trie T.
    Step 1: FOR i = 1 TO h DO:
        IF $\text{SONPOINTER}_v(D_i^K) \neq$ NULL THEN v $\leftarrow$ $\text{SONPOINTER}_v(D_i^K)$ ELSE GO TO
        STEP 2;
        IF i = h THEN key has been found and algorithm should immediately
        terminate by returning the pointer v. END OF STEP 1.
    Step 2: Recall that D was defined to equal:
        i) the least digit greater than $D_i^K$ satisfying $\text{SONPOINTER}_v(D) \neq$ NULL
           (if such a digit exists);
        ii) and otherwise the greatest digit less than $D_i^K$ satisfying
           $\text{SONPOINTER}_v(D) \neq$ NULL.
    Calculate the value of D by performing the obvious binary search
    inside INNERTREE(v).
    Step 3: If $D > D_i^K$ THEN advance to the address LOWKEY($\text{SONPOINTER}_v(D)$)
    (where one suitable CLOSEMATCH key can be retrieved) ELSE retrieve
    CLOSEMATCH(K) from the address HIGHKEY($\text{SONPOINTER}_v(D)$).
    END OF PROGRAM.
```

first case, it will retrieve SUCCESSOR(K) by advancing to the position LOWKEY(SONPOINTER$_v$(D)). In the latter case, it will retrieve PREDECESSOR(K) by advancing to the analogous position HIGHKEY(SONPOINTER$_v$(D)).

Our formal algorithm for performing CLOSEMATCH(K) retrievals is illustrated in Figure 2. It is easy to see that the three steps of this algorithm have respective runtime complexities of $\Theta(i)$, $\Theta(\log b)$, and $\Theta(1)$. Thus, CLOSEMATCH retrievals have a total worst-case complexity $\Theta(h + \log b)$.

The above result implies that p-fast tries have an *overall* worst-case retrieval complexity $\Theta(h + \log b)$. This is because the needed runtimes for PREDECESSOR, SUCCESSOR, and SUBSET queries are obtained with two-part procedures, that first perform a CLOSEMATCH retrieval and then use the pointers from the trie's ordered list.

We will now briefly explain the significance of such a complexity $\Theta(h + \log b)$ for overall retrieval. Recall that our original goal was to manipulate a set S of nonnegative integer keys, all of which are less than some initially specified bound M. If we take $h = \left\lceil \sqrt{\log M} \right\rceil$ and $b = \left\lceil 2^{\sqrt{\log M}} \right\rceil$ then a p-fast trie of size (h,b) will provide us with a worst-case overall-retrieval complexity $\Theta(\sqrt{\log M})$ in worst-case memory space $O\left(N \cdot \sqrt{\log M}\; 2^{\sqrt{\log M}}\right)$ for searching the set S.

This result is a significant improvement over stratified trees [5,7,8, 9] in applications where $M \gg N$ because its memory space is smaller than the counterparts $\Theta(M \log\log M)$ and $\Theta(M)$ for stratified trees. Van Emde Boas has mentioned that a retrieval complexity $\Theta(\sqrt{\log M})$ is possible for an alternative data structure which has a branching factor 2^d for all nodes at depth d. Although [7] attributed memory space $\Theta(M)$ to this data structure, it is not difficult to see that the better upper bound $O(N2^{\sqrt{2 \log M}})$ is possible when $M \gg N$. This quantity is not as good as the memory space $O(N \sqrt{\log M}\; 2^{\sqrt{\log M}})$ of p-fast tries.

We can also produce substantial further reductions in memory space by pruning the bottoms of fast tries. In the context of stratified trees, Van Emde Boas [8] has observed that pruning will reduce worst-case memory space from $\Theta(M \log\log M)$ to $\Theta(M)$. Now we will explain how a similar idea, applied to fast tries, produces an even greater improvement that reduces memory space to $\Theta(N)$ without any increase in asymptotic retrieval time.

One preliminary definition must be introduced before we can define a q-fast trie. Let S denote an initial set, S* an ordered set of keys: $0 = K_0^* < K_1^* < K_2^* \ldots < K_L^* = M$, and S_i the subset $\{K \in S\colon K_{i-1} \le K < K_i\}$. Then, the set S* will be said to form a c-partition of set S iff each set S_i has a cardinality between c and 2c.

A q-fast trie of size (h,b,c) will be defined as a two-part data structure for representing a set S. Its first section will be a p-fast trie T of size (h,b) that represents some set S*, which forms c-partition of S. Its second part will be a forest of 2-3 trees whose i-th tree T_i represents the set S_i. The leaf for key K_i^*, in trie T, will contain a pointer to the corresponding tree T_i.

It is not hard to see how to attain an overall worst-case retrieval complexity $\Theta(h + \log b + \log c)$ for q-fast tries of size (h,b,c) by using the retrieval complexity $\Theta(h + \log b)$ of p-fast tries and the logarithmic retrieval property of 2-3 trees [1]. Taking $h = \lceil \sqrt{\log M} \rceil$, $b = \lceil 2^{\sqrt{\log M}} \rceil$, and $c = hb$, it is apparent that a worst-case retrieval time $\Theta(\sqrt{\log M})$ is possible in memory space $\Theta(N)$ with the use of q-fast tries.

The full length version of this paper [10] discusses this topic in greater detail and proves that insertions and deletions also have the same worst-case time complexity $\Theta(\sqrt{\log M})$. Johnson [3], while investigating a different problem, observed that the technique of [10] could be usefully applied to reducing memory space. We conjecture that the overall worst-case retrieval complexity $O(\sqrt{\log M})$ of q-fast tries is the optimal result in worst-case space $O(N)$. A curious fact is that the overall worst-case retrieval complexity $O(\log \log M)$ is achievable in expected space $O(N)$ under the uniform probability distribution [11]. Yao's lower bound [12] appears relevant to our conjecture, as does the Tarjan-Yao upper bound [6] for the special case of MEMBER queries in a static environment.

ACKNOWLEDGEMENTS

I would like to thank Y. Edmund Lien, Daniel D. Sleator, and Eric Wolman for their very helpful suggestions on presentation.

REFERENCES

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Mass., 1974.

[2] E. Fredkin, "Trie Memory," Comm. ACM, 3 (1962), pp. 490-499.

[3] D. B. Johnson, "A Priority Queue in which Initialization and Queue Operations Take O(log log D) Time," Report No. CS 81-13, Penn. State University, July, 1981.

[4] D. E. Knuth, The Art of Computer Programming, Vol. 3: Sorting and Searching, Addison-Wesley, Reading, Mass., 1973.

[5] D. E. Knuth, widely disseminated classroom notes on stratified trees.

[6] R. E. Tarjan and A. C. Yao, "Storing a Sparse Table," Comm. ACM, 22 (1979), pp. 606-611.

[7] P. Van Emde Boas, "Preserving Order in a Forest in Less than Logarithmic Time," Proceedings of the 16-th Annual Symposium on Foundations of Computer Science, (1975), pp. 75-84.

[8] P. Van Emde Boas, "Preserving Order in a Forest in Less than Logarithmic Time and Linear Space," Information Processing Lett., 6 (1977), pp. 80-82.

[9] P. Van Emde Boas, R. Kaas, and E. Zijlstra, "Design and Implementation of an Efficient Priority Queue," Math. Systems Theory, 10 (1977), pp. 99-127.

[10] D. E. Willard, "New Trie Data Structures which Support Very Fast Search Operations of Order $\sqrt{\log M}$," manuscript, June, 1981.

[11] D. E. Willard, forthcoming report.

[12] A. C. Yao, "Should Tables Be Sorted?," Journal ACM, 28 (1981), pp. 615-628.

Analysis of a Randomized Data Structure for Representing Ordered Sets[1]

Jon Louis Bentley
Departments of Computer Science and Mathematics
Carnegie-Mellon University
Pittsburgh, PA 15213

Donald F. Stanat
Department of Computer Science
University of North Carolina
Chapel Hill, NC 27514

J. Michael Steele
Department of Statistics
Stanford University
Stanford, CA 94305

Abstract -- Janko has described an efficient randomized sorting algorithm for ordered sets. In this paper we extend the basic data structure of his algorithm to a randomized data structure for representing ordered sets, and give a precise combinatorial analysis of the time required to perform various operations. In addition to a practical data structure, this work gives a new probabilistic lower bound and an instance of a problem whose randomized complexity is provably less than its deterministic complexity.

1. Introduction

In this paper we will consider the problem of maintaining a set from a totally ordered domain under the operations Member, Insert, Delete, Predecessor, Successor, etc. The basic data structure we will use to represent such a set is due to Janko [1976, 1977]: a sorted linked list implemented by the two arrays `Link[0..N]` and `Value[1..N]` (where `Link[0]` points to the first element in the list).

In Section 2 we will define the problem more precisely and see that the worst-case complexity of performing a Member search is linear in N. In Section 3 we will see a randomized algorithm for Member searching that requires only $\sim 2N^{1/2}$ comparisons, and in Section 4 we will see a lower bound that shows that $\sim(2N)^{1/2}$ comparisons are necessary, on the average. Those sections deal only with Member searching; other operations on sets are considered in Section 5. Finally, conclusions are offered in Section 6.

---

[1] This research was supported in part by the Office of Naval Research under Contract N00014-76-C-0370 and in part by the National Science Foundation under Grant MCS-78-07736.

2. The Problem and Its Deterministic Complexity

The data structure we will study was briefly mentioned in the introduction; we will now describe it in more detail. It is a sorted linked list implemented in contiguous storage by the two arrays `Link[0..N]` and `Value[1..N]`. The pointer `Link[0]` points to the first element of the list, `Value[Link[0]]`; the next element can be found in `Value[Link[Link[0]]]`, and so forth. The end of the list is denoted by an element whose `Link` field contains -1. Furthermore, we will insist that the array is *dense*: `Value[1..N]` must contain N elements of the represented set. The sortedness of the linked list implies that if `Link[I]` is not -1, then

`Value[I]` ≤ `Value[Link[I]]`.

We will often refer to `Value[I]` and `Link[I]` together as node `I`. The following figure illustrates the array representation of the sorted linked list <2.6, 3.1, 4.1, 5.3, 5.8, 5.9, 9.7>.

`I`	0	1	2	3	4	5	6	7
`Value[I]`		3.1	4.1	5.9	2.6	5.3	5.8	9.7
`Link[I]`	4	2	5	7	1	6	3	−1

It is clear that performing a Member search in such an array requires accessing at most N elements of the array (either by following `Link` fields through the list or simply by iterating through `Value` fields of the array).

We will now show that in the worst case, linear time is necessary to locate whether a given element is in the list. We will assume that a (deterministic) search algorithm is composed of operations of the following types.

1. Determine the index of the node at the head of the list (by accessing `Link[0]`). There is one operation of this type.
2. Determine the `Value` of node `I`, for $1 \leq I \leq N$ (by accessing `Value[I]`). There are N operations of this type.
3. Determine the successor of node `I`, for $1 \leq I \leq N$ (by accessing `Link[I]`), where node `I` was previously accessed by an operation of Type 1, 2 or 3. (There are N operations of this type, but only r of them are valid when r nodes have been accessed by previous operations.)

We will assume that each of the above operations has unit cost. (Note that if operations of Type 3 have no cost then binary search can be used to solve the problem in logarithmic time.)

Our model assumes that a protagonist specifies a sequence of the above operations while an adversary ensures that N operations will be required. We will assume that the adversary knows the value of the key the protagonist seeks, which we will call `V`, and that other key values may be assigned arbitrarily by the adversary. We will describe a strategy that enables the adversary to delay returning `V` until the protagonist has specified a sequence of N operations. The argument is a typical adversary argument, and is omitted from these proceedings for brevity.

3. A Randomized Algorithm

In this section we will consider the randomized complexity of the searching problem. We will investigate an algorithm derived from Janko's [1976]; it uses the fact that the list is stored in contiguous storage to sample randomly several elements of the sorted list, and then follows the links from the one that is closest to the desired element in a sequential scan. Janko analyzed this method statistically using approximations; we will study a precise combinatorial analysis showing that if we sample k elements then the number of comparisons made in the sequential scan is $\sim N/(k+1)$. Choosing k as $N^{1/2}$ shows that $\sim 2N^{1/2}$ comparisons suffice on the average to find the desired element.

The algorithm we will study is described in the following pseudo-Pascal program that searches for the element `T` in the data structure described in the previous section.

```
P := 0
for I := 1 to K do
    J := Random(1,N)
    if Value[J] < T and Value[J] > Value[P] then
        P := J
while Value[Link[P]] < T do
    P := Link[P]
if Value[Link[P]] = T then
    (* T is in Link[P] *)
  else
    (* T is not present; we could insert it after P *)
```

Program 1. Search for `T` in a linked list.

The operation of the above program is straightforward. The `for` loop randomly samples k elements of the array, and stores in `P` the pointer to the node of greatest `Value` among all sampled nodes with `Value` less than `T`. (The function `Random(1,N)` returns an integer chosen uniformly from the range 1..N.) After that, the `while` loop follows `Link` fields until it either finds `T` or an element greater than `T`; the appropriate processing of the element could be inserted in place of the comments in the `if` statement. The above code assumes that `Value[0]` is $-\infty$ (this is the zeroth node in the list) and that `Value[-1]` is ∞ (this is the last node in the list); those assumptions are not necessary for the algorithm, but they do make the code simpler and more efficient.

We will turn now to an analysis of the running time of Program 1. If k is an input parameter to the procedure, then the only variable in the running time is the number of comparisons made by the `while` loop. That number is a function of two conditions: the position of `T` in the list, and the random numbers returned by the random number generator. Our analysis will be worst-case in one sense: we will assume that the search object `T` is greater than every element in the list (that is, than all elements of `Value[1..N]`). The most important part of our analysis will be probabilistic: we will assume that all N^k possible sequences of outputs from the random number generator are equally likely. Notice that no assumptions are made regarding the distribution of the elements of `Value[1..N]` or of their order in the array.

The following lemma reduces the analysis of Program 1 to a straightforward combinatorial problem. The lemma assumes the definition that $M_{k,N}$ denotes the minimum of k integers chosen uniformly with replacement from 1..N.

Lemma 3.1: Reduction to minimum.

The distribution of the number of comparisons made by the `while` loop in Program 1 to find a search value `T` that is greater than all elements of `Value[1..N]` is identical to the distribution of $M_{k,N}$.

Proof:

The key to this lemma is the permutation function *Perm* defined as follows for any valid assignment of `Link` values (*Perm* is bijective on the integers in 1..N). Let j be the unique integer such that `Link[`j`]` $= -1$; then $Perm(j) \triangleq 1$. Note that if the variable `P` in Program 1 is j after the `for` loop, then the `while` loop will make exactly one comparison. In general, if $Perm(i) = k$ and `Link[`m`]`$=i$, then $Perm(m) \triangleq k+1$. It is easy to establish by induction the fact that if the variable `P` in Program 1 contains j after the `for` loop, then the `while` loop will make $Perm(j)$ comparisons. Furthermore, the `for` loop will set `P` to be that j with minimum value of $Perm(j)$. Thus *Perm* can be viewed as mapping array indices into their distances from the end of the linked list, and Program 1 chooses the minimum index of the k it samples.

To establish the lemma we use the fact that choosing a random integer from 1..N and applying an arbitrary permutation function gives a random integer from 1..N.

The above lemma has reduced the analysis of Program 1 to the combinatorial problem of analyzing the distribution of $\mathrm{M}_{k,N}$; the next lemmas do exactly that.

Lemma 3.2: Expectation of minimum.

The expected value of the minimum of k elements chosen randomly with replacement from 1..N is

$$\mathrm{E}[\mathrm{M}_{k,N}] = \Big(\sum_{1 \le k \le N} i^k \Big) \;/\; N^k \,.$$

Proof:

The key observations in the proof are that there are a total of N^k k-arrangments of 1..N, and that $(N-i+1)^k$ of those arrangements have a minimum value of i. The above expectation then follows by straightforward algebra.

Using standard approximations with the above lemma shows that for fixed k and large N, we have $\mathrm{E}[M_{k,N}] \sim N/(k+1)$. However, we will be concerned with the case that k varies as a function of N (and specifically, when k grows as $N^{1/2}$); we therefore need the following lemma.

Lemma 3.3: Expectation of minimum for growing k.

For integer $N > 2$ and real a such that $N^{1/2}a \ge 2$,

$$N^{1/2}/a - 1/2 \;\le\; \mathrm{E}[\mathrm{M}_{N^{1/2}a,N}] \;\le\; N^{1/2}/a \;+\; 1/a \;+\; 2/(a^3 N^{1/2}).$$

Proof Sketch:

The proof is by the Euler-McLaurin formula, and is omitted from these proceedings for brevity.

With the above lemmas in hand, it is easy to establish the following theorem.

Theorem 3.4: Analysis of Program 1.

If the variable K in Program 1 is equal to $N^{1/2}$, then the program makes $2N^{1/2} + O(1)$ accesses of the `Value` array.

Proof:

Immediate from Lemmas 3.1 and 3.3.

4. A Lower Bound on the Randomized Complexity

Janko [1976] proved that the $\sim 2N^{1/2}$ comparisons of the algorithm of Section 3 are optimal over the class of all strategies that first perform a number of random probes and then sequentially search from the best. In this section we will show a weaker lower bound on the randomized complexity of searching a linked list that holds in a more robust model. We will first define a probabilistic game such that the number of operations in the game is exactly the number of values accessed by the searching algorithm, and then prove a lower bound on the number of operations in the game.

We will now study two variants of a probabilistic game; the second variant corresponds exactly to the problem of searching in a linked list, while the first version contains more information than the searching problem.

Definition 4.1: Probabilistic games G_1 and G_2.

Both variants G_1 and G_2 of this game involve two integers, i and N; the value of N remains fixed throughout the game, and the value of i is originally N. The goal of the player is to reduce the value of i to zero in the minimal expected number of steps. The value of the game G_1 is defined to be the expected number of steps used by an optimal strategy and is denoted by $G_1(N)$; similarly the value of the game G_2 is denoted by $G_2(N)$. A step consists of performing one of the following two operations.

- D (for Decrement): Replace i by $i-1$.
- G (for Guess): Choose j to be an integer uniform from $1..N$ and replace i by j if $j \leq i$.

There are two variants of the game: in the first variant, G_1, the player knows at all times the current value of i. In the second variant, G_2, i is unknown and the player is notified when i becomes zero.

The game G_2 models the problem of searching a linked list of N elements if we assume only the operations of randomly probing an element of the list (corresponding the the G operation) and following a link (corresponding to D). The initial condition (that $i=N$) corresponds to having a pointer to the initial element of the list, and the final condition assumes that the search element is greater than all elements in the list.

The analysis of Section 3 shows that in game G_2 (where i is unknown), the strategy of performing $N^{1/2}$ G operations followed by a series of D operations performs a total of $2N^{1/2}+O(1)$ operations, on the average. This establishes the upper bound that $G_2(N) \leq \sim 2N^{1/2}$. We conjecture that this bound is tight, but we cannot prove that. In the rest of this section we will prove a lower bound for G_2 by proving a lower bound on G_1; since the game G_1 contains more information than G_2, a lower bound on the former implies a lower bound on the latter (that is, $G_1(N) \leq G_2(N)$). Specifically, we will demonstrate an optimal strategy for G_1 that shows that $G_1(N) = \sim(2N)^{1/2}$. These results together identify the value of $G_2(N)$ to within a constant factor of $2^{1/2}$.

We will use dynamic programming to establish the optimal strategy for the game G_1. We will let the function $F(i,N)$ denote the number of operations required by the optimal strategy for G_1 for given values of i and N. The following theorem establishes a recurrence for $F(i,N)$.

Theorem 4.2: A dynamic programming recurrence.
The function $F(i,N)$ satisfies the dynamic programming recurrence

$$F(i,N) \; = \; 1 + \min\{\, F(i-1,N),\; 1/N) \cdot \sum_{1 \leq j < i} F(j,N) \; + \; (N-i+1) \cdot F(i,N)/N \,\},$$

with the boundary condition that $F(0,N) = 0$.

Proof:
The boundary condition states that the problem has been solved once i is zero. The recursive part minimizes the expected value for arbitrary values by choosing the operation with minimum expected cost. If the D operation is chosen, the value of i is decreased by one. If the G operation is chosen, there are two possible outcomes. With probability $(N-i+1)/N$, the value of i is unchanged; with probability $1/N$, it assumes each of the $i-1$ positive integers less than i. The recursion is valid by the principle of optimality.

Note that the optimal strategy corresponding to the recurrence must know the value of i to choose which operation to perform at each step.

To express the solution to the above recurrence, we will let T_k denote the k^{th} triangular number, $k \cdot (k-1)/2$.

Theorem 4.3: Solution of the recurrence.

If $N = T_m + k$, where $0 \leq k < m$, then

$$F(i,N) = F(i, T_m + k) \; = \begin{cases} i & \text{for } i \leq m \\ m + k/m & \text{for } i > m \end{cases}$$

Note that T_m is the largest triangular number not greater than N.

Proof:
The proof for a fixed value of N is a straightforward inductive argument on i, and is omitted in these proceedings for brevity.

Because $T_m \sim m^2/2$, the value of m in Theorem 4.3 is $\sim(2N)^{1/2}$. This establishes the claim that $G_1(N) = F(N,N) = \sim(2N)^{1/2}$.

The solution to the dynamic programming recurrence implies the optimal strategy for the game G_1. Let m be the largest triangular number less than N (as in Theorem 4.3); if the current value of i is greater than m, then the optimal strategy performs a G, otherwise it performs a D. It is interesting to observe that both the solution to the recurrence and the optimal strategy remain unchanged if the game is modified so that the G operation always sets i to the random integer j.

5. A General Data Structure

In previous sections we have considered only the problem of searching the linked list to determine if it contains a given element. It is easy to perform many other set operations on this structure; the following list summarizes those operations, and describes their costs in terms of the number of `Value` elements accessed.

- **Member:** The previous sections studied the problem of searching to determine whether a given element is a member of the set represented by the linked list. *Cost:* $\sim 2N^{1/2}$.
- **Insert:** A new element can be inserted in the list by using Program 1. *Cost:* $\sim 2N^{1/2}$.
- **Delete:** The first step in deleting an element is to find that element by a search algorithm like that in Program 1, and then modify the `Link` field of its predecessor to point to its successor; that step makes $\sim 2N^{1/2}$ references to the `Value` array. The next step must patch the "hole" created in the dense array by moving the last element of the array to the vacant position; searching for the last element requires $\sim 2N^{1/2}$ references. *Cost:* $\sim 4N^{1/2}$.
- **Predecessor:** The element immediately preceding a given element can be found by a modification of Program 1. *Cost:* $\sim 2N^{1/2}$.
- **Successor:** The element immediately succeeding a given element can be found by a modification of Program 1. *Cost:* $\sim 2N^{1/2}$.
- **Minimum:** The minimum element in the set is pointed to by `Link[0]`. *Cost:* 1.
- **Maximum:** The maximum element in the set can be found by a modification of Program 1. *Cost:* $\sim 2N^{1/2}$.

Each of the above operations is straightforward to implement given the model of Program 1 and basic techniques for dealing with data structures for searching (described, for example, by Knuth [1973]). Furthermore, the simplicity of that code implies that the constant factors of the running time of the program will be relatively slow (Janko [1976] describes an implementation in detail). The only deviation that a programmer should make from the code of Program 1 deals with the random number generation: since many random number generators are very expensive, it might be preferable to use some other approach to sample the k elements.

This data structure is superior to all of those described in Knuth [1973] for certain applications; the salient attributes of such applications are listed briefly.

- Space is important. This structure uses only one extra word of storage per element, while binary search trees use at least two extra words, and various hashing schemes use varying amounts of extra storage. However, the storage for this structure must be available in a single contiguous block.
- The "orderedness" operations of Successor, Predecessor, Min and Max are frequent; these

are not possible in most hashing schemes.

- Insertions and deletions are frequent. If the data structure changes rarely, binary search in a sorted array is very efficient.
- Program simplicity is important. Each operation on this structure requires only about a dozen lines of code, while some operations on balanced binary search trees require over one hundred lines of code.
- Run time is important for problems of medium size (where medium means that N is between, say, 100 and 10000). If N is below that range, simple sequential strategies are probably efficient enough. If N is above that range, then the logarithmic search time of binary search will be necessary for many applications. When N is in that range, though, the low constant factors of this structure will make it competitive with binary search trees.

6. Conclusions

The following list summarizes our extensions to Janko's work.

- We have extended the problem from sorting an ordered set to maintaining an ordered set under various set operations. The data structure has a number of desirable properties both theoretically and practically (it uses little space, has good randomized time and is easy to implement).
- We have tightened Janko's upper bounds on Member searching to give an exact combinatorial analysis of the procedure.
- We have given a lower bound on the problem of searching a linked list with random probes that is weaker than Janko's but holds in a more robust model.
- We have interpreted Janko's construction as a randomized algorithm; that class of algorithms was identified as a class only after Janko's [1976] paper was published. This problem provides an example of a problem whose randomized complexity is provably less than its deterministic complexity; few examples of such problems are currently known.

This work raises a number of open problems. Perhaps the most obvious is to tighten the bounds on the randomized complexity of the searching problem (using the game G_2); the bounds we present are a factor of $2^{1/2}$ apart. We conjecture that the game G_2 of Section 4 has the value of $\sim 2N^{1/2}$. A broader open problem is to explore additional randomized data structures, and the application of other probabilistic games to the analysis of randomized algorithms and data structures.

References

Aho, A. V., J. E. Hopcroft and J. D. Ullman [1974]. *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA.

Janko, W. [1976]. "A list insertion sort for keys with arbitrary key distribution," *ACM Transactions on Mathematical Software 2*, 2, June 1976, pp. 143-153.

Janko, W. [1977]. "An insertion sort for uniformly distributed keys based on stopping theory," *International Computing Symposium 1977*, April 1977, North-Holland Publishing, pp. 373-379.

Knuth, D. E. [1973]. *The Art of Computer Programming, volume 3: Sorting and Searching*, Addison-Wesley, Reading, Massachusetts.

DELETING THE ROOT OF A HEAP

ERNST E. DOBERKAT*
Department of Mathematics and Computer Science
Clarkson College of Technology
Potsdam, New York 13676

ABSTRACT

Using a continuous model, the average behavior of the familiar algorithm for root deletion is considered, when every heap has the same probability to occur. The analysis centers around the notion of a viable path in the tree representation, i.e. such a path the new label of the root may be allowed to travel when the heap is reconstructed. In case the size of the heap is a power of 2 it is shown that both the expected number of comparisons and of interchanges are asymptotically equal to the respective numbers in the worst case.

1. INTRODUCTION

In this paper some aspects of root delection of a heap are investigated from an expected performance point of view. This algorithm is well known and works as follows: suppose you have a heap of size N such that the label of the root is the smallest element in this heap. If you want to extract this smallest element, then you will copy it elsewhere, and the rightmost node that has no son will be placed at the root. Since the heap property is violated in doing this, this element will percolate the tree down, until it has reached a position in which the heap property is reconstructed. This process implies that the second smallest element is now on top of the heap, and if the process of deleting the root and reconstructing the heap property is repeated until all elements in the heap are processed, the elements of the heap are arranged in ascending order. Thus root deletion is at the heart of the selection phase of heapsort, and of the use of heaps as priority queues.

It will be assumed that the heap that serves as an input to the algorithm is chosen at random in the sense that every heap has the same probability to occur. It turns out to be advantegeous to assume that the elements in the respective heaps are real numbers (rather than integers in the range of 1,...,N), and to make use of a continuous model (rather than to work by counting permutations). This is mainly due to the fact that in a continuous model the Change of Variables Formula ([5], 6.148) is available, and this tool makes the analysis somewhat easier. The main idea in this analysis is to characterize those paths in the heap, the element replacing the originally given root of the heap may travel in order to reconstruct the heap property; call such a path a viable one. After having done this characterization, it turns out to be necessary to compute the probabilities with which the element in question does indeed follow such a viable path. These probabilities may be computed explicitly in terms of the binary expansion of N, and some intrinsic properties of the viable path. Using these probabilities and the length of the paths, the expected numbers of interchanges, and of comparisons necessary for the algorithm are computed. This is done in case N is a power of 2, say $N = 2^n$, and in this case the following is shown: the expected number of interchanges equals $n - 1 + O(n/N)$, and the expected number of comparisons equals $2n - 1 + O(n/N)$. Note that the worst case is n, resp. 2n.

---

*Some of this work was done at the Fernuniversität, Hagen, West Germany

The plan of the paper is as follows: in Section 2, some preparations are done, in particular some anatomical details of heaps are quoted from the corresponding chapter in [4], the algorithm is given in a more formal manner, and some probabilistic assumptions are made explicit. Section 3 deals with distributional aspects, it is shown how root delection affects randomness (or uniform distribution), and how the corresponding distribution looks like. In Section 4 the expectations are calculated by means of computing the probabilities mentioned above for the general case, and then we specialize to $N = 2^n$. Calculating the quantities in question requires a little bit of a formal apparatus, the provision of which would be beyond the page limits. The interested reader is invited to consult [3] for proofs, and for a broader outline.

2. PREPARATIONS

A N-dimensional real array b[1...N] is said to be a *heap* iff $b[\lfloor i/2 \rfloor] < b[i]$ holds for any i such that $2 \leq i \leq N$. If $\{1,\dots,N\}$ is represented as a tree such that 1 is the root, and $\lfloor i/2 \rfloor$ is the father of node i, and if furthermore b[i] serves as a label for node i, then b is a heap iff no node has a smaller label, than its father. The tree representation will be important in the sequel, and some notation will be used with respect to it.

To begin with, if $1 \leq i \leq N$, let $g(N,i)$ denote the size of the subtree rooted at i and let $G(N,i)$ be the product of the sizes of the subtrees dominated by i. A *special node* in the tree is a node which is on the path from N to the root 1; define $t(n,0) := N$, and $t(N,i+1) := \lfloor t(N,i)/2 \rfloor$ for $0 \leq i < \lfloor \log_2 N \rfloor$, then $\{t(N,i);\ 0 \leq i \leq \lfloor \log_2 N \rfloor\}$ constitutes the *special path*.

Let N have the binary representation $(1b_{n-1}\ldots b_0)_2$, then it is easily seen that $t(N,i) = (1b_{n-1}\ldots b_i)_2$ holds, and from [4], Exercise 5.2.3.20, it is derived that we have $g(N,t(N,i)) = (1b_{i-1}\ldots b_0)_2$. The following algorithm will be investigated in the sequel:

Algorithm Rootdeletion:

```
Input:       a heap b[1...N]
Output:      x, the smallest element of the heap, and the heap b[1...N-1]
             which is the originally given heap with its root deleted and
             the heap property reconstructed
Method:      1.  [Copy the root]
                 x ← b[1]; b[1] ← b[N];
             2.  [Reconstruct]
                 i ← 1;
                 loop:  if i is no leaf and a son of i contains a greater
                        label then i does
                        then
                            k ← son of i with the smallest label;
                            b[i] ↔ b[k];
                            i :=k;
                            goto loop
                 endif.
```

This algorithm is a variation of Floyd's method to construct heaps ([1], p.90), and if repeated N times, it returns the array sorted in descending order. The question which will be investigated below is the expected number of interchanges and comparisons that are necessary in the execution of this algorithm. It will be assumed that every heap has the same probability to occur, but rather than doing so for permutations, we take the heaps to be chosen at random from

$$H(N) := \{(x_1,\ldots,x_N);\ 0 \le x_i \le 1, \text{ the vector is a heap}\}.$$

Then to say that every heap has the same probability to be considered as an input to the algorithm means that for any Borel set $A \subset H(N)$ $\mathrm{Prob}(A) := \chi_N.\lambda^N(A)$ defines the probability in question. Here $\chi_N := G(N,1)$ denotes the product of all subtree sizes, and λ^N is N-dimensional volume or, more technical, Lebesgue measure restricted to the Borel sets of H(N). The equality above derives from [2], Theorem 1 and reflects the fact that heap creation by means of Floyd's method preserves uniform distribution.

It should be noted that the discrete case in which permutations are delt with is implicitly contained in the continuous case considered here, see [3].

We will describe the way b[N] percolates the tree by a word $v \in \{0,1\}^*$, where 0 means that it is interchanged with a left son, and 1 denotes an interchange with a right son. Let for such a v be v_j the prefix of length j, for the sake of completeness v_0 is defined as the empty word e. Given such a word v, let Z(v) be the endnode of the path represented by v.

3. DISTRIBUTIONAL ASPECTS

Let for $v \in \{0,1\}^*$ as the path to be followed, H(N,v) be the set of all $x \in H(N)$ such that the N-th component x_N does indeed follow v. Hence $x \in H(N,v)$ holds iff $x_N > \min\{x_{Z(v_{j-1}0)}, x_{Z(v_{j-1}1)}\} = x_{Z(v_j)}$ holds for $1 \leq j \leq |v|$. Thus if $x \in H(N,v)$, then x_N will be interchanged with $x_{Z(v_j)}$ for $1 \leq j \leq |v|$, but it will not be interchanged with any son of $x_{Z(v)}$, provided there is one. Now call v a *viable path* for x_N iff $H(N,v) \neq \emptyset$, and denote the set of all viable paths by $\mathcal{V}_N$. Note that not every path in the tree corresponding to {1,...,N} starting at 1 can be a viable path for N, since any part $b_{n-1}\dots b_j$ of the special path excluding $\lfloor N/2 \rfloor$ *must not* be a viable path; $b_{n-1}\dots b_1$ constitutes a viable path, and it is easily seen that $v \in \mathcal{V}_N$ holds iff both $Z(v) \leq N - 1$, and $v \notin \{b_{n-1}\dots b_i;\ 2 \leq i \leq n-1\}$ are true. Note that H(N) equals the disjoint union of H(N,v) for $v \in \mathcal{V}_N$. Denote the action of the algorithm by T_N, hence T_N maps H(N) into G(N-1), where $G(N-1) := \{(x_1,\dots,x_{N-1},y);\ (x_1,\dots,x_{N-1}) \in H(N-1),\ 0 \leq y \leq x_1\}$. In order to describe the distributional effects of T_N, some subsets of G(N-1) will be needed; these subsets will depend on viable paths for N and will shown to be the bijective images of the sets of the form H(N,v). To begin with, let $G(N-1, b_{n-1}\dots b_1) := G(N-1)$, and if v is a viable path which does not stop in $\lfloor N/2 \rfloor$, then $G(N-1,v) := \{(x_1,\dots,x_{N-1},y) \in G(N-1);\ x_{Z(v)} > x_{\lfloor N/2 \rfloor}\}$ is defined. Consider for example the case N = 6, and X := (1/5, 1/3, 1/4, 1/1, 1/2, 1/6), then $X \in G(5,v)$ for v = 0, 00, 01, 1. It will be seen in a moment that X lies in the image of exactly the classes H(N,0), H(N,00), H(N,01), and H(N,1) under T_N:

3.1 Lemma: Given $v \in \mathcal{V}_N$, $T_N: H(N,v) \to G(N-1,v)$ is a bijection. □

At the first glance it is not clear what connection between 3.1 and the task of analyzing the algorithm may be constructed. An immediate consequence is the following: given $z \in G(N-1)$, the number M(z) of $v \in \mathcal{V}_N$ with $z \in G(N-1,v)$ counts the number of possible heaps which are mapped by T_N to z. From this observation the distribution of T_N may be derived:

3.2 Proposition: Given a Borel set $A \subset G(N-1)$, let $\mu(A) := \mathrm{Prob}(T_N \in A)$ be the probability that the output of the algorithm is an element of A. Then

$$\mu(A) = \chi_N \cdot \int_A M(z)dx$$

holds. □

Denote by $T(x)$ the number of interchanges which are necessary to reconstruct the heap property when the root of $x \in H(N)$ is deleted. Thus $T(x)$ equals $|v|$ whenever $x \in H(N,v)$, and we have for the expectation $\mathbb{E}(T)$ that it equals $\chi_N . \Sigma\{|v|\lambda^N(H(N,v));\ v \in V_N\}$. If $S(x)$ is the number of comparisons which are done in reconstructing the heap property for $x \in H(N)$, and if $x \in H(N,v)$ for $v \in V_N$, such that $Z(v)$ has i sons $(0 \leq i \leq 2)$ in the tree corresponding to $\{1,\ldots,N-1\}$, then $S(x) = 2|v| + i$ holds. Thus

$$\begin{aligned}\mathbb{E}(S) = 2.\mathbb{E}(T) &+ b_0.\chi_N.\lambda^N(H(N,b_{n-1}\ldots b_1)) + \\ &+ 2.\chi_N . \Sigma\{\lambda^N(H(N,v));\ Z(v) \text{ has two sons} \\ &\qquad\qquad \text{w.r.t. } \{1,\ldots,N-1\}\}.\end{aligned}$$

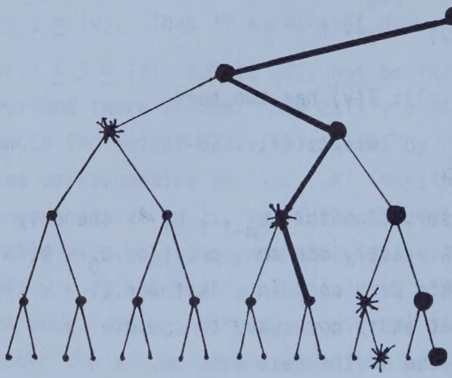
The second summand comes from the observation that $b_{n-1}\ldots b_1$ is the only viable path which ends in a node with exactly one son, provided $b_0 = 1$; in case $b_0 = 0$, i.e. if N is even, this path ends in a leaf w.r.t. $\{1,\ldots,N-1\}$. Thus it is apparent that it is necessary to compute $\lambda^N(H(N,v))$ for $v \in V_N$. This will be done in the next section.

4. COMPUTING THE EXPECTATIONS

Fix for the moment a viable path $v \in V_N$. Then from Lemma 3.1 and from Proposition 3.2 it is clear that $\lambda^N(H(N,v)) = \lambda^N(G(N-1,v))$ holds. Let $K(N-1,v)$ be the set of all (N-1)-heaps in which the label of node $Z(v)$ is greater than the label of $\lfloor N/2 \rfloor$; for the sake of completeness $K(N-1,b_{n-1}\ldots b_1) := H(N-1)$ is defined, and it is not difficult to see that we have $\lambda^N(G(N-1,v)) = \lambda^{N-1}(K(N-1,v))/N$. Hence the task of computing $\lambda^N(G(N-1,v))$ is reduced to that of computing $\lambda^{N-1}(K(N-1,v))$. In case $v = b_{n-1}\ldots b_1$ (or $b_{n-1}\ldots b_1 0$, provided N is odd) this is done easily, since the respective sets equal $H(N-1)$, which has χ_{N-1}^{-1} as its (N-1)-dimensional volume. In the general case, however, some hairy computations will be needed. We will proceed as follows: to begin with, we fix labels on the special path with respect to N, but excluding N, and on the nodes of v. With these fixed labels in mind we have a look at the volume of all heaps which may have these labels at the respective nodes; then we will integrate over all possible values which may be fixed and obtain in this way the (N-1)-dimensional volume of $K(N-1,v)$, by an application of Fubini's theorem ([5], 6.121, p. 352).

Let us have a look at a possibility to obtain a heap labeling in

$K(N-1,v)$, when the special path $\{t(N,i); 1 \leq i \leq n\}$ (excluding N, of course) has already a label. First, we cut the path $\{t(N,i); 1 \leq i \leq n\}$ out of the tree representing $\{1,\dots,N-1\}$ and are left with a forest of smaller trees. These trees are all rooted at the brothers of the special nodes $t(N,n-1),\dots,t(N,1)$, and of $t(N,0)$, if N is odd. The following figure might help to clarify things (the nodes $t(N,1),\dots,t(N,n)$ are marked by heavier dots, the respective brothers by asterisks, and the path v in question by heavy lines).



Now consider a tree in this residual forest with the property that no part of v is in it. This tree may be labeled as a heap under no other constraint, than that the label of its root is not smaller, than the label of its father (in the 'big' tree). All trees but one are labeled in this manner. The remaining one contains that part of the path v which lies not on the special path. In order not to come in conflict with the heap property the root, is labeled as are the roots of the other trees, viz., with a number not smaller than the label of its father. But in order not to come into conflict with the requirement that the heap thus produced in a member of $K(N-1,v)$, the endnode is assigned a number that is not smaller, than y_1. Having in mind these restrictions and repeating the considerations from above, this yields a labeling of this subtree, too.

Returning to our fixed path v, let us note some characteristica associated with it, viz.,

- $P(v)$ as the smallest index j such that $b_{n-1}\dots b_j$ is a prefix of v,
- $\rho(v)$ as the number of nodes which are on v but not on the special path $b_{n-1}\dots b_0$,
- $R(v)$ as the number of nodes which are on the path from the endnode of v to a leaf including that endnode.

Hence we see that $R(v) = P(v) - \rho(v) + b_{p(v)-1}$. The viable path v may be written as $v = b_{n-1}\dots b_{P(v)}\; d_1\dots d_{\rho(v)}$, where $d_1 \neq b_{P(v)-1}$. Note that the case $P(v) = n$ is not excluded. Moreover $\sigma(v) := R(v) + \rho(v) - 1$ is the number of nodes in a path starting from $(1b_{n-1}\dots b_{P(v)}d_1)_2$ to a leaf (including its start node), thus the tree rooted at $(1b_{n-1}..b_{P(v)}d_1)_2$ has $2^{\sigma(v)} - 1$ nodes. Consider as an example $N = 52$ and the path $v = 1001$: we have $n = 5$, $P(v) = 3$ (since 10 is a prefix of v, but 101 is not), $R(v) = 2$, $\sigma(v) = 3$, $\rho(v) = 2$. With these values in mind a general expression for the measure of the set in question may be derived (see [3], p. 21-29). Specializing to $N = 2^k$, noting that $\sigma(v) = P(v) - 1 + b_{P(v)\,-\,1}$, and multiplying this measure by χ_N/N, we see that for $p(r,s)$ as the probability that x_N follows a path v having the characteristics $\sigma(v) = s$ and $\rho(v) = r$, the equality

$$p(r,s) = N^{-1} \prod_{i=1}^{n} 2^i \cdot \prod_{i=1}^{r} (2^{s+1-i}-1) \cdot \prod_{j=s+1}^{n} (2^j-1)^{-1} \times$$

$$\times\; 2^{-(s+1)\cdot(r-1)} \cdot \sum_{a=1}^{r} \Big[\prod_{j=0}^{s} (2^{s+1-a} + 2^j - 2) \prod_{\substack{t=1\\ t\neq a}}^{r} (2^{-t}-2^{-a}) \Big]^{-1}$$

holds, since $b_i = 0$, $0 \leq i \leq n - 1$. There are 2^{r-1} paths with these characteristics and any of them has length $n - s - 1 + r$. Thus the expected number $\mathbb{E}(T)$ of interchanges equals

$$\mathbb{E}(T) = \sum_{s=1}^{n-1} \sum_{r=1}^{s} 2^{r-1}\cdot(n-s-1+r)\cdot p(r,s) + O(\tfrac{1}{N})$$

since r may take values between 1 and s, and s may range from 1 to n-1. The $O(N^{-1})$ term takes into account that parts of the special path with respect to N that are viable paths.

Evaluating these sums, the divided difference turns out to cause some trouble, see [3], 31-34, 37-39. It finally comes out that we have:

4.1 Theorem: In case $N = 2^k$, rootdeletion requires $n-1+O(n/N)$ interchanges on the average. □

Let us turn our attention to the number S of necessary comparisons. The characterization of $\mathbb{E}(S)$ at the end of Section 3 tells us that we should have a look at those paths which end in a node that has exactly two sons with respect to the tree representing $\{1,\dots,N-1\}$. No such path v will have the characteristic $\sigma(v) = 1$ or $\rho(v) = \sigma(v)$. With this in mind, one derives from 4.1 the following

Corollary: In case $N = 2^k$, rootdeletion requires $2n-1+O(n/N)$ comparisons on the average. □

REFERENCES

[1] Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, Mass., 1974.

[2] Doberkat, E.E.: Some Investigations on the Average Performance of Heapsort. 21st FOCS, Syracuse, N.Y., 1980, 229-237.

[3] Doberkat, E.E.: Deleting the Root of a Heap. Preprint, 1981.

[4] Knuth, D.E.: The Art of Computer Programming, Vol. 3 - Sorting and Searching. Addison-Wesley, Reading, Mass., 1973.

[5] Stromberg, K.E.: An Introduction to Classical Real Analysis. Wadsworth International, Belmont, Calif., 1981.

ON THE AVERAGE-CASE COMPLEXITY OF 'BUCKETING' ALGORITHMS

SELIM G. AKL AND HENK MEIJER
Queen's University, Kingston, Ontario, Canada

ABSTRACT

We analyze the average-case behavior of a general class of algorithms to which the method of buckets can be applied. In particular, we give sufficient conditions under which an algorithm *A* that solves a problem *P* of size n in $O(n^k)$, $k>1$, worst-case time, can be made to run in linear expected time.

INTRODUCTION

Let *P* be a problem defined on a set of reals $\{x_1, x_2, ..., x_n\}$ and let A_1 be an algorithm that solves *P* in $O(n^k)$ steps in the worst-case for some constant $k>1$. Now consider the following algorithm.

This work was supported by the Natural Sciences and Engineering Research Council of Canada under grant NSERC - A3336 and by a scholarship from the Ontario Graduate Scholarship program.

Algorithm A_2

Step 1: Partition the set $\{x_1,x_2,...,x_n\}$ into subsets (or *buckets*) $b_1,b_2,...,b_n$, such that all the elements of b_i are strictly smaller than all the elements of b_{i+1}, for $i=1,2,...,n-1$.

Step 2: Apply A_1 to each bucket.

Step 3: Merge the partial solutions to obtain the global one. □

Algorithm A_2 is a special kind of divide-and-conquer known as the method of buckets (or 'bucketing' for short). It has been used to obtain good average-case performances in solving such problems as sorting [5,6,10,15], searching [15] and selection [1]. To date all analyses of A_2 were dependent on the particular problem to which it was applied. In this paper we derive a general result concerning the expected complexity of A_2. As steps 1 and 3 usually take linear time, the rest of this paper will be concerned with the complexity of step 2.

AVERAGE-CASE ANALYSIS

Our analysis rests on two assumptions:

(i) the input to A_2 is $\{X_1,X_2,...,X_n\}$, where the X_i's are independent identically distributed random variables with common density f;

(ii) the bucket membership of an input element is determined using the function

$$\text{bucket}\ (X_i) = \left\lceil \frac{X_i - U}{V - U} \cdot n \right\rceil \qquad \text{for } U < X_i < V$$

where $\lceil \cdot \rceil$ denotes the ceiling function, assumed to require constant time,

$U = \min \{X_1, X_2, \ldots, X_n\}$

and $V = \max \{X_1, X_2, \ldots, X_n\}$.

Let $D = V - U$ and let C denote the complexity of step 2. The expected value of C, denoted by $E(C)$, is given by our first theorem.

Theorem 1

If $\sup\{f(x)\} < \infty$,

then

$E(C) = O(n\ E(D^{k-1}))$.

Proof:

Let us define the quantity p_j for $j = 1, 2, \ldots, n$, by

$$p_j = \text{Prob} \left\{ \frac{X_i - U}{V - U} \in \left(\frac{j-1}{n}, \frac{j}{n}\right) \,\middle|\, U,V \right\} \tag{1}$$

i.e. p_j is the probability that element X_i falls in bucket j.

(Note that $\sum_{j=1}^{n} p_j = 1$.)

Further, let

$\tilde{F}(x) = \text{Prob}\{X_i \le x \mid U,V\}$

and

$S = \sup \{\frac{d}{dx}\ \tilde{F}(x)\} = \frac{\sup\ \{f(x)\}}{F(V) - F(U)}$

where the supremum is taken over all values of x for which the derivative exists, and

$F(x) = \int_{-\infty}^{x} f(z)\, dz$.

From (1) we have

$$p_j = \text{Prob}\ \{X_i \in (a,b) \mid U,V\}$$

for some interval (a,b) of width D/n. It follows that

$$\begin{aligned} p_j &= \tilde{F}(b) - \tilde{F}(a) \\ &\le S\ (b - a) \\ &= \frac{SD}{n} \quad . \end{aligned}$$

Now, let B_j be a random variable representing the number of elements in bucket j after step 1. The B_j's have a binomial distribution with parameters n and p_j .

Using the moment generating function for the binomial distribution B(n,p), namely,

$$E(e^{xt}) = (p\ e^t + (1-p)\ )^n$$

we get

$$E(B_j^k\ ) = O(\max(1, n^k\ p_j^k)) \quad .$$

Therefore

$$\begin{aligned} E(C \mid U,V) &= E(\sum_{j=1}^{n} B_j^k \mid U,V) \\ &= \sum_{j=1}^{n} O(n^k\ p_j^k) \\ &\le \frac{n}{SD}\ O((SD)^k) \\ &= n\ O(D^{k-1}) \end{aligned}$$

since $S \to \sup\{f(x)\}$ as $n \to \infty$.

Thus

$$\begin{aligned} E(C) &= E(n\ \ O(D^{k-1})) \\ &= O(n\ \ E(D^{k-1})) \quad . \ \square \end{aligned}$$

SOME SPECIAL CASES

The result in theorem 1 describes the average-case behavior of A_2 as a function of n, k and D regardless of the problem P being solved, provided that the algorithm is applicable to it. In order to express E(C) more precisely, we need to know the expected value of D^{k-1}. The two following corollaries are examples.

Corollary 1

If f has compact support (i.e. D is bounded by a constant) then $E(C) = O(n)$. □

Corollary 2

If f has exponentially vanishing tails, then for k=2 and 3,

$$E(C) = O(n \log_2^{k-1} n)$$

Proof

Since $E(D) = O(\log_2 n)$
and $\mathrm{Var}(D) = O(1)$ [9]
the result follows. □

As a final special case, let A_1 have a worst-case complexity of $O(n \log_2 n)$. We have

Theorem 2

If $\sup\{f(x)\} < \infty$, then
$E(C) = O(n \log_2 E(D))$.

Proof

Since [12]

$E(X \log X) \le (E(X^2)\ \ E(\log_2^2(2+X)))^{1/2}$ by the Cauchy-Schwarz inequality

$\le (E(X^2))^{1/2} \log_2 E(2+X)$ by Jensen's inequality, it follows that

$$\begin{aligned} E(C|U,V) &= E\left(\sum_{j=1}^{n} (B_j \log_2 B_j) \middle| U,V\right) \\ &\le \frac{n}{SD} \ O(SD \log_2(2+SD)) \\ &= O(n \log_2(2+D)) \end{aligned}$$

Thus

$$E(C) = O(n \log_2 E(D)) \quad . \quad \square$$

Corollary 3

If f has exponentially vanishing tails, then

$$E(C) = O(n \log_2 \log_2 n) \quad . \quad \square$$

This result is particularly relevant to the sorting problem [10].

OPEN PROBLEMS

We have been concerned in this paper with the average-case behavior of the method of buckets when applied only to one-dimensional problems. It should be pointed out, however, that bucketing has been used successfully to solve problems in higher dimensions (i.e. when each x_i in the input is a vector rather than a scalar). Examples include closest-point problems [3,7,11,14], diameter problems [4], the largest-gap problem [8] as well as finding convex hulls [2,13]. As in the one-dimensional case, all existing analyses of bucketing algorithms in higher dimensions are a function of the problem being solved. Performing a general average-case analysis of the method of buckets in dimensions higher than one, remains an interesting and challenging problem.

REFERENCES

[1] Allison, D.C.S. and Noga, M.T., Selection by distributive partitioning, Information Processing Letters, Vol. 11, No. 1, 1980, pp. 7 - 8.

[2] Bentley, J.L., Faust, M.G. and Preparata, F.P., Approximation algorithms for convex hulls, Technical Report CMU-CS-80-109, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[3] Bentley, J.L., Weide, B.W. and Yao, A.C., Optimal expected-time algorithms for closest-point problems, ACM Transactions on Mathematical Software, Vol. 6, No. 4, 1980, pp. 563 - 580.

[4] Brown, K. Q., Geometric transforms for fast geometric algorithms, Technical Report CMU-CS-80-101, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

[5] Devroye, L. and Klincsek, T., Average time behaviour of distributive sorting algorithms, Computing, Vol. 26, No. 1, 1981, pp. 1 - 7.

[6] Dobosiewicz, W., Sorting by distributive partitioning, Information Processing Letters, Vol. 7, No. 1, 1978 pp. 1 - 6.

[7] Fortune, S. and Hopcroft, J., A note on Rabin's nearest-neighbor algorithm, Information Processing Letters, Vol. 8, No. 1, 1979, pp. 20 - 23.

[8] Gonzalez, T., Algorithms on sets and related problems, Department of Computer Science, University of Oklahoma, 1975.

[9] Gumbel, E.J., Statistics of Extremes, Columbia University Press, New York, 1958.

[10] Meijer, H. and Akl, S.G., The design and analysis of a new hybrid sorting algorithm, Information Processing Letters, Vol. 10, No. 4, 1980, pp. 213 - 218.

[11] Rabin, M., Probabilistic algorithms, in: Algorithms and Complexity, Traub, J., Ed., Academic Press, 1976, pp. 21 - 40.

[12] Rudin, W., Real and Complex Analysis, McGraw-Hill, New York, 1966.

[13] Toussaint, G.T. and Akl, S.G., The Convex Hull Problem, Plenum Press, New York, to appear.

[14] Yuval, G., Finding nearest neighbours, Information Processing Letters, Vol. 5, No. 3, 1976, pp. 63 - 65.

[15] Weide, B.W., Statistical methods in algorithm design and analysis, Technical Report CMU-CS-78-142, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

QUERY PERFORMANCE OF A MANY-DIMENSIONAL BEST-MATCH ALGORITHM

MICHAEL R. DUNLAVEY
Boston College
Chestnut Hill, MA

ABSTRACT

A best-match algorithm is presented which is suitable for problems, such as spelling correction, in which a) the number of dimensions is more than a few, and b) queries are not random. An asymtotic upper bound for query performance, in the context of an artificial model, is shown to be $O(h(\log N)^h)$ where h is the edit distance. Numerical results with the model yield many insights into the performance of the algorithm.

INTRODUCTION

The best known formulation of the best-match problem is by Minsky and Papert(6). Given an N-word data base of b-bit words, where $b>\log_2 N$, the problem is to find the nearest matching word to a given key, where "nearest" is defined as having the smallest Hamming distance. They regarded the chances of finding a good worst case algorithm, i.e. one needing less than $O(2^b)$ space and less than O(N) time, for large b, as particularly gloomy. Nevertheless, several people have since devised practical algorithms for restricted variations of the problem.

Elias proposed a hash-coding method based on the existence of perfect codes. Rivest(8) showed that this algorithm was, in a sense, optimal. What was left unsaid was how to optimize the choice of a code, what the quantitative performance of a given code would be, or how to devise perfect codes of the necessary size and length.

Bentley, Friedman, and Finkel(2)(5) have examined the best-match problem in Cartesian spaces of low dimensionality (small b). Their method is essentially a depth-first search coupled with branch-and-bound. They achieve query performance in time $O((\log N)^b)$.

Aho(1) and Wagner(9) have examined best-match in context free and regular languages, respectively. This can be related to best-match table search by modeling the table of N words by a FSA of at least N states. Wagner's algorithm achieves query times in $O(bN^2)$, thus favoring small databases (grammars) but long words.

The literature on Information Theory discusses Sequential decoding (Fano(4)). The idea of Sequential Decoding is to represent the data base (code) as a FSA. Instead of having one program counter, the state of the decoder is represented as a set of program counters, ordered by their probability of correctness. As each digit of the input word is received, the set of program counters is updated. In the literature, it is generally assumed that the redundancy of the code (b/logN) is small, since in coding theory, code efficiency is paramount. It is also assumed that the space reserved for the set of program counters is small and can overflow. Thus, concern is placed on how often the decoder makes a mistake. So, if we have large b/logN and working storage is not a problem, we must look elsewhere for performance results.

Nillsson(7) discusses the problem of search in a FSA, except that it is called a heuristic search tree. This search, as in Sequential Decoding, relies on a list of states in working storage. This list is kept ordered by estimated cost f(s). At each step, the least costly state is removed from the list, and its sons, with their estimated costs, are added to the list. As long as these estimated costs increase monotonically from father to son, the minimum cost leaf nodes will be found first. However, in performance analyses of heuristic search, it is usually assumed that f(s) is not chosen conservatively. That is, it includes a term based upon a heuristic rule which can end up being wrong. Thus, it can find a good answer faster at the expense of not being guaranteed to find the best answer first.

The method we present here is similar to heuristic search in a FSA, but using a very conservative cost function f(s). It is also very similar to Sequential Decoding. The restrictions we place on the problem are those which are relevant in a practical problem such as spelling correction. Namely, query keys are not randomly generated, rather the number of errors per word is typically small. The length of words is not so great that their sheer length increases the likelihood of errors by a troublesome amount. Coding redundancy b/logN is typically around two or three. Under these conditions, the query time of the algorithm is dominated by the error distance h, and an upper bound of $0((h+1)(k-1)^{h}(1+\log_{k}N)^{h+1})$ is derived, where k is the arity of the alphabet.

THE ALGORITHM

The data base FSA is represented as a k-ary trie, where k is the arity of the input alphabet. The key is an array key(j), j=1..b, each digit in the range 0..k-1. There is a queue of elements, each containing a node and an integer j, indicating a digit position in the key. When a leaf node is encountered, it is printed in some meaningful way. There are two procedures, *descend* and *top*. *Descend* is a normal trie search algorithm, except that it queues all the rejected siblings. *Top* just keeps on applying *descend* to nodes in the queue until the queue becomes empty, or a leaf node is printed.

```
procedure descend(node,j);
  begin
    if terminal(node) then write(node)
    else if node not equal nil then begin
      descend(node(key(j)),j+1);
      enqueue(node,j)
      end
    end;
procedure top;
  begin
    descend(root,1);
    while not empty(queue) do begin
      dequeue(node,j);
      for i:=1 to k-1 do
        descend(node((key(j)+i) mod k), j+1);
      end
    end;
```

The technique for extending this algorithm to handle insertions, deletions, and transpositions of digits is simple enough and need not be discussed here.

PERFORMANCE ANALYSIS

To estimate the performance of the algorithm the method used is this:

1) Invent a model pseudo-random k-ary trie of known structure.
2) Using counting arguments, derive an exact expression for $R(b,h)$, the number of trie nodes visited during a search.
3) Derive an upper bound on R. Numerically evaluate R versus other independent parameters.

The construction of the model is based on the observation that tries representing randomly generated tables have a particular structure. If a given node has i ancestors above it, the expected number of leaves under it is N/k^i. Thus the trie will be bushy out to about the $\log_k N$-th rank, below which it will be stringy. The model we define is called an a,b-trie, and it is bushy for a ranks, after which it is stringy for (b-a)more ranks. Each stringy part simply represents the digits of the alphabet cycled over and over, starting with 0. An example is shown in Figure 1. The counting argument, and the algorithm itself, has no predilection for any particular query key, so we can analyse it with any key of our choosing, and we choose the key of all zeros, in which case Hamming distance becomes Hamming weight. We can then twist the drawing of the trie, as shown in Figure 2, so that each node falls in a layer which reflects the number of its ancestral links representing nonzero digits. We define the Hamming weight of a node as the level of the layer in which it lies. The algorithm simply walks the trie by layers.

Defn. An a,a-trie is a k-ary trie of a+1 ranks where each nonterminal node has k sons, indexed 0..k-1.

Defn. An a,b+1-trie is derived from an a,b-trie by giving each terminal node one son, under the index (b-a) mod k.

Defn. $a = \log_k N$ = entropy of data base (assuming equiprobability).

Defn. h = edit distance = number of digits not matching key.

Thm. The number of nodes in the trie is:

$$M = \frac{k^{a+1}-1}{k-1} + (b-a)k^a$$

Thm. The number of leaves of weight h in an a,a-trie is:

$$P(h) = \binom{a}{h}(k-1)^h$$

and an upper bound on this is $(k-1)^h a^h$

Thm. The number of nodes of weight h in an a,b-trie is:

$$Q(b,h) = \text{nodes in bushy part} + \text{nodes in stringy part}$$

$$= \binom{a+1}{h+1}(k-1)^h + \mathrm{Sum}_{i=1}^{i=(b-a)} P\left(h-\left\lfloor \frac{i(k-1)}{k} \right\rfloor\right)$$

and an upper bound on this is

$$(a+1)^{h+1}(k-1)^h + (b-a)a^h(k-1)^h$$

Thm. The number of nodes of weight h or less in an a,b-trie:

$$R(b,h) = \mathrm{Sum}_{i=0}^{i=h} Q(b,i)$$

and the upper bound is:

$$(h+1)(k-1)^h((a+1)^{h+1}+(b-a)a^h)$$

When $a=\log_k N$ is substituted into the final bound, it is:

$$(h+1)(k-1)^h \left[(\log_k N +1)^{h+1} + (b-\log_k N)(\log_k N)^h\right]$$

This bound is gross, as it quickly exceeds the number of nodes in the trie. It is an open problem to derive a tighter bound.

NUMERICAL RESULTS

The above theorems were used to calculate R and M at various values of h for arities of k=2,4,8,16 and table sizes $N=2^{12}$,24,48,96. Figure 3 shows the error correcting ability of the algorithm. Note that the vertical scale is logarithmic. It shows that the entropy of the table is a kind of barrier. It also shows that with larger k, this barrier may be approached more closely. In fact, as k approaches infinity, the shape of the curve approaches the integral of a square pulse.

Figures 4 and 5 depict R versus N at various values of error distance. The scales have been chosen so that powers of logarithmic performance appear as straight lines of various slopes. It can be seen that over most of the data performance is approximated by $(\log_k N)^{h+1}$. However, the curves necessarily change their shape when they come into close proximity to the M curve, which is the number of nodes in the trie.

Finally, if one were about to design such a trie, he would be interested in the proper choice of k. This choice depends on the mechanism by which errors occur, as shown in Figure 6. The context corresponds to a teletype transmission problem, where each character is sent as a byte of binary digits. Mechanism a) assumes that errors occur on the transmission line, but infrequently, so that an erroneous bit results in an erroneous byte. In this case, it is better to use a binary trie. Mechanism b) assumes that errors are made by the typist. If a character is mistyped, half of that character's bits will be wrong, on the average. In this case, it is better to use a character-oriented trie. Mechanism c) assumes that errors occur on the transmission line at a Poisson rate, so the length of a code word b might be a factor in determining the error distance of a given received word. As in a), it is better to use a binary trie.

RELATED PROBLEMS

The technique of this algorithm can be modified in various ways. For example, it can be made to work in a linearly ordered table, just like binary search. What price is paid for this is not known. Another structure is a hybrid trie/linear table, where the trie is used merely as an index into the table. In such a case, it might be that the cost of determining edit distance to a specific word of the table is considered a small constant because it uses special hardware. How can the algorithm be modified to take advantage of this, and what savings would arise? The algorithm can also be applied in Cartesian spaces, with a little modification, but its performance relative to other algorithms is unknown.

REFERENCES

1. Aho, A.V., Peterson, T.D., A Minimum-Distance Error-correcting Parser for Context-Free Languages. SIAM J. Comp. Vol 1, No 4, December 1972.

2. Bentley, J.L., Multidimensional Divide and Conquer, Communications of the ACM, Vol 23, No 4, April 1980.

3. Dunlavey, M.R., A Logarithmic Best-Match Algorithm for Symbolic Applications, Boston College Report 81-40, School of Management.

4. Fano, R.M., A Heuristic Discussion of Probabilistic Decoding. IEEE Trans. Info, Theory, Vol IT-9, April 1963.

5. Friedman, J.H., Bentley, J.L., Finkel, R.A., An Algorithm for Finding Best Matches in Logarithmic Expected Time. ACM Trans. on Mathematical Software, Vol 3, No 3, September 1977.

6. Minsky, M.L., Papert, S., Perceptions. MIT Press, Cambridge Mass., 1969, pp 222-225.

7. Nillsson, N.J., Principles of Artificial Intelligence, Tioga Press, Palo Alto, CA 1980.

8. Rivest, R., On the Optimality of Elias' Algorithm for Performing Best Match Searches, in Information Processing 74, North-Holland Publishing Company, Amsterdam, 1974, pp 678-681.

9. Wagner, R.A., Order-n Correction of Regular Languages, Comm. of the ACM, Vol 17, No 5, May 1974.

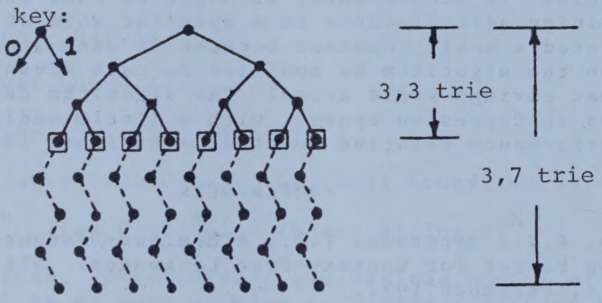
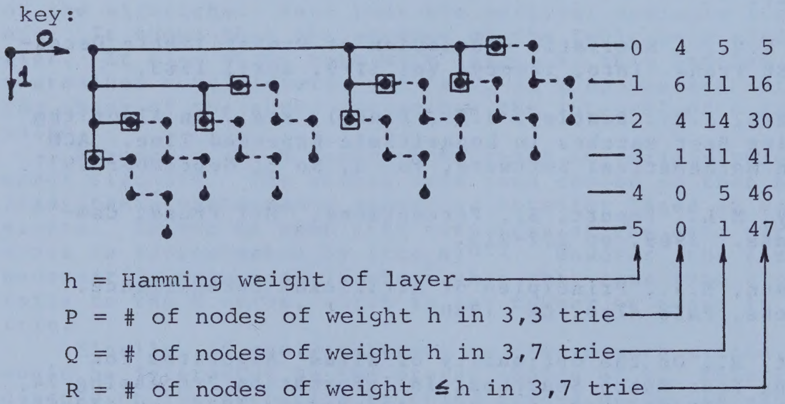


Fig. 1. Example construction of an a,b trie.



h	P	Q	R
0	4	5	5
1	6	11	16
2	4	14	30
3	1	11	41
4	0	5	46
5	0	1	47

h = Hamming weight of layer

P = # of nodes of weight h in 3,3 trie

Q = # of nodes of weight h in 3,7 trie

R = # of nodes of weight ≤ h in 3,7 trie

Fig. 2. Illustration of counting nodes by Hamming weight in a,b trie.

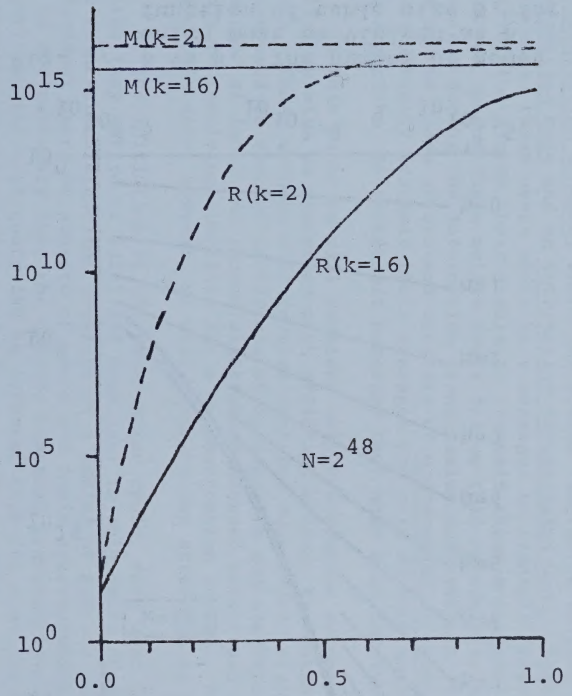


Fig. 3. R vs h/a. The number of nodes which must be visited as a function of error distance h.

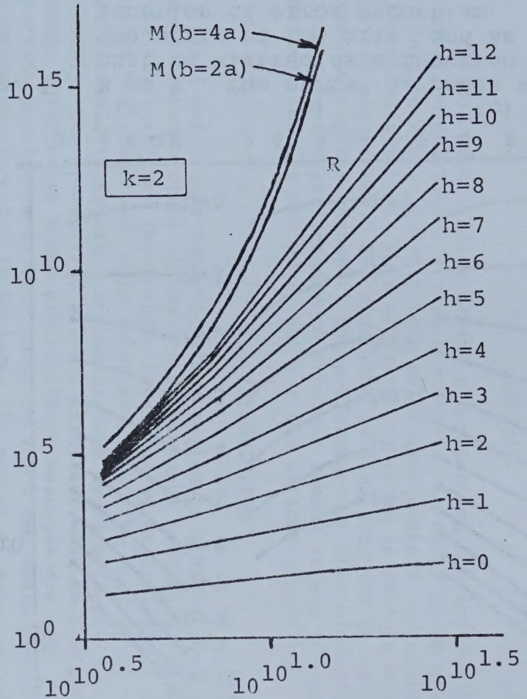


Fig. 4. R vs N. The number of nodes which must be visited as a function of table size N, for a binary trie (k=2).

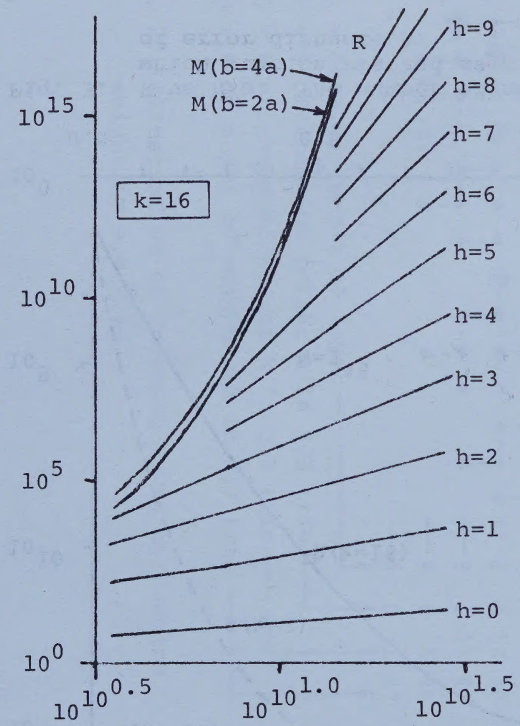


Fig. 5. R vs N. The number of nodes which must be visited as a function of table size N, for a 16-ary trie (k=16).

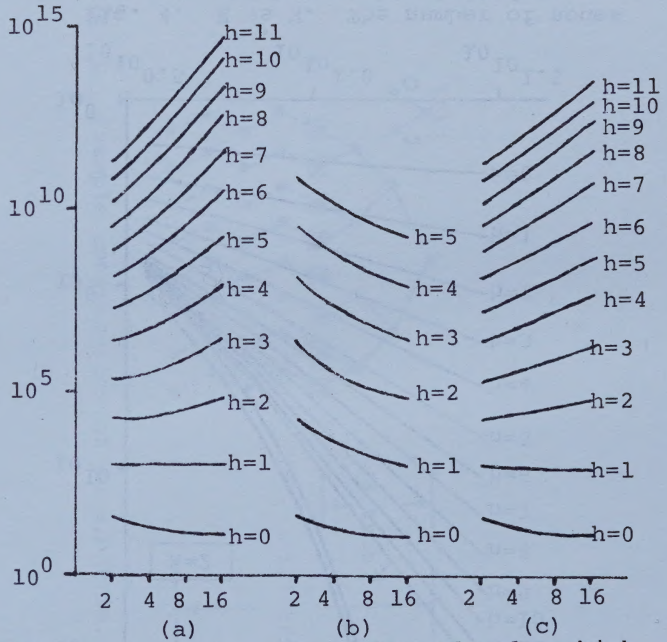


Fig. 6. R vs k. The number of nodes which must be visited as a function of k, the arity of the trie, and as a function of error mechanism.

AN APPLICATION OF GROUP TESTING TO THE DESIGN OF MULTI-USER ACCESS PROTOCOLS

DONALD F. TOWSLEY AND JACK KEIL WOLF
Department of Electrical and Computer Engineering, University of
Massachusetts, Amherst, Massachusetts 01003

ABSTRACT

In this paper, ideas behind group testing are applied to the design of reservation multi-user access protocols. An early group testing algorithm is shown to apply directly to a multi-user channel typically associated with polled systems. This algorithm is shown to perform better than other reported algorithms. The algorithm is also extended to apply to the random access multi-user channel. The performance of this algorithm also appears better than the performance of previous random access reservation protocols. Numerical results are presented for the various algorithms.

I. INTRODUCTION

In a multi-user communications system, a community of users shares a common communications channel to interchange messages. In such a system, it is the norm that the traffic generated by an individual user is sporadic so that any given user only requires use of the channel at infrequent and unpredictable intervals. If two or more users attempt to transmit simultaneously, the transmitted signals interfere with each other and no valid message interchange takes place. (Such messages are said to collide.) An important problem is to find means by which the users can schedule their transmissions in order to maximize the number of messages carried by the system.

Two classes of systems have been considered. The first class consists of reservation schemes, where the system fluctuates between two modes, a reservation mode and an information transmission mode. During the reservation mode, those users which have messages to transmit, make known to the rest of the system their desire to transmit these messages. A schedule is then set up and these active users follow this schedule to transmit their messages during the information transmission mode without interference. The second class of systems consists of nonreservation schemes where the users directly transmit their messages and rely on some contention resolving algorithm to retransmit messages that have been involved in collisions.

In this paper we will consider only reservation schemes and specifically, we will be concerned with efficient methods for making these reservations. These methods will involve conflict resolution algorithms much like those used to resolve collisions of messages in nonreservation schemes. As a matter of fact, the second algorithm discussed here could equally well apply to resolving conflicts for a nonreservation scheme. For simplicity and consistency, however, we will restrict our attention to reservations.

At the initiation of the reservation cycle, each user is in one of two states (say active or idle) and each user is only aware of its own state. The reservation algorithm consists of a series of queries addressed to individual users or subsets of users and their corresponding responses. It is not necessary that the queries be actual transmissions since in some algorithms the next query can be inferred from the previous responses or alternatively the queries may be fixed a priori. The reservation cycle is complete when the states of all users are determined. We will refer to this general technique as polling (although the term probing has been used to describe algorithms where subsets of users are simultaneously queried).

The simplest, but least efficient, polling algorithm is roll-call polling where individual users are queried in some fixed order. An alternative view of this strategy is that each user is permanently assigned a fixed reservation slot and transmits a signal in its slot when it wishes to make a reservation.

Hayes [HAY 78] and Gudjohnsen et al [GUD 80] described polling algorithms where groups of users are queried simultaneously. These algorithms differ from each other in the form of the responses to the queries and how several responses combine on the channel. They have the common characteristic of choosing the subsets of users to be queried as subtrees in a binary tree. The Gudjohnsen algorithm is closely related to the Capetanakis [CAP 79] algorithm originally proposed for resolving conflicts in a non-reservation scheme.

In this paper we employ the methodology of group testing to develop improved versions of both the Hayes and Gudjohnsen (or Capetenakis) algorithms. Group testing was introduced by Sobel and Groll [SOB 59] as a method of improving the efficiency of blood testing. Dorfman [DOR 43] reported a procedure used in testing the blood samples of volunteers and draftees during World War II. A fixed number of blood samples was pooled and tested for a particular disease. A negative result indicated that all persons whose blood was contained in the pool was free of the disease. A positive result indicated one or more diseased persons in the pool at which point in the Dorfman procedure the blood sample of each person in the pool was tested individually. Sterrett [STE 57] improved upon the Dorfman procedure by returning to pooled sampling after the individual tests located a single diseased sample. Sobel and Groll further generalized this idea by testing smaller and smaller subsets of pooled samples rather than immediately going to individual testing when a pooled sample gave a positive result. An important aspect of the Sobel and Groll group testing procedure was that the optimum sample size was chosen at each step in the algorithm. They compared their results with several sub-optimum procedures, one of which involved halving the pool size when a defective sample was located. This is exactly the procedure suggested by Hayes except they realized that some of the queries in the Hayes scheme could be eliminated because they are redundant. Indeed the numerical results given by Sobel and Groll agree with those obtained for the Hayes scheme after the redundant questions are eliminated.

In the next section we review the results of Sobel and Groll as they apply to the Hayes model. Then in the following section we apply group testing to improve upon the Gudjohnsen (or Capetanakis) model. We will find that these new algorithms contain the following improvements over the original algorithms.

1. The algorithms can be implemented on a system containing an arbitrary number of users. The tree search algorithms are only defined for systems containing a number of users that is a power of two.
2. The new algorithms are simpler to implement.
3. The new algorithms exhibit a slight improvement in performance.

We make the following assumptions about the activity of the users. We assume N users, each user having a probability p of being active and probability q=1-p of beind idle. The random vaiables describing the activity of the N users are assumed to be statistically independent of one another. Thus, prior to the polling cycle, the probability of having exactly i active users is given by the binomial formula

$$\binom{N}{i} p^{i} q^{N-1}, \quad i = 0, 1, \ldots, N.$$

II. GROUP TESTING FOR THE HAYES MODEL

Let S denote the set of N users. When a subset of S is queried, the response to the query is a 1 if and only if one or more users in the group are active; otherwise the response is a 0. A response of 0 identifies the condition that all users in that group are idle. A response of 1, however, will only completely identify the status of the users in the group if the group contains a single user.

The algorithm contains the following features. At any point in time there exist three sets of users. The first set, U, contains those users for which nothing is known (unknown set). The second set, A, is known to contain at least one active user (active set). Finally, the third set, K, contains all users that have been served (known set). Whenever the active set is nonempty, the algorithm queries a subset of it. If this subset contains at least one active user, the remainder of the active set is recombined with the unknown set. The subset then becomes the new active set. This last recombination feature does not appear in the tree search algorithms. A statement of the Sobel and Groll algorithm follows. ($\emptyset$ denotes the empty set.)

```
0.  U := S; K := $\emptyset$; A := $\emptyset$;

1.  if U = $\emptyset$ then goto 3
                 else query a subset $X \subsetneq U$
                      if response = 0
                         then K := K + X;
                              U := U - X;
                              goto 1
                      if response = 1
                         then A := X;
                              U := U - X;
                              if A contains one user
                                 then K := K+A;
                                      goto 1
                                 else goto 2

2.  query a subset $X \subsetneq A$
    if response = 0
       then K := K + X;
            A := A - X;
            goto 2
    if response = 1
       then U := U + A - X;
            A := X;
            if A contains one user
               then K := K+X;
                    goto 1
               else goto 2

3.  end
```

Note that in the above algorithm the size of the subset is chosen to minimize the remaining number of queries.

The recombination aspect of the Sobel and Groll algorithm is incorporated in the step U := U + A - X which takes the untested portion of the set A, A - X, and recombines it with U after X, the tested portion of A, is found to have a positive response. The justification for such a step is that the number of active users in A - X can be proved to have a binomial distribution, the same distribution which describes the number of active users in U.

In order to determine the efficiency of this algorithm, Sobel and Groll derived a set of recurrence relations which express the expected number of queries for a group of N users, H(N), in terms of the expected number of queries for groups of n users, n = 0, 1, ..., N-1 (and other quantities). The complete relations are:

$$H(n) = 1 + \min_{1\le x\le n} \left(q^x H(n-x) + (1-q^x)G(x,n)\right)$$

$$G(x,n) = 1 + \min_{1\le y\le x-1} \left\{ \left(\frac{q^y-q^x}{1-q^x}\right) G(x-y,n-y) + \left(\frac{1-q^y}{1-q^x}\right) G(y,n) \right\}, \quad n\ge x\ge 2$$

with boundary conditions

$$H(0) = 0,\ G(1,N) = H(N-1) \text{ for } N = 1, 2, \ldots\ .$$

Here G(x,n) is the expected number of queries required conditioned on the event that the active set A contains x users and the unknown set U contains n-x users. The results of solving this equation numerically are given in Table I along with the corresponding results for the original Hayes algorithm (shown in [ ]) and an improved Hayes algorithm which eliminates redundant queries (shown in ( )).

Table I

Expected Number of Queries for Hayes Model

p	.1	.2	.3	.4	.5
N					
2	[1.38](1.29)	[1.72](1.56)	[2](1.81)	[2](2)	[2](2)
	1.29	1.56	1.81	2	2
3	1.66	2.25	2.75	3	3
4	[2.49](2.11)	[3.44](3.07)	[4](3.62)	[4](4)	[4](4)
	2.05	3.01	3.60	4	4
5	2.49	3.74	4.51	5	5
6	2.94	4.44	5.39	6	6
7	3.37	5.19	6.29	7	7
8	[4.98](4.14)	[6.88](6.14)	[8](7.24)	[8](8)	[8](8)
	3.76	5.91	7.17	8	8
9	4.16	6.63	8.06	9	9
10	4.57	7.37	8.95	10	10
11	4.99	8.10	9.84	11	11
12	5.42	8.82	10.72	12	12
13	5.86	9.55	11.61	13	13
14	6.31	10.28	12.5	14	14
15	6.77	11.00	13.39	15	15
16	[9.96](8.28)	[13.76](12.28)	[16](14.48)	[16](16)	[16](16)
	7.22	11.73	14.28	16	16

III. GROUP TESTING FOR GUDJOHNSEN (OR CAPETANAKIS) MODEL

Again let S denote the set of N users. Users are assumed to indicate their active status by responding to a query with their address. Each user's address is assumed unique so that a single address transmitted as a response to a query identifies the respondent. However, two or more addresses are assumed to collide. The received signal is either the null signal (when no users respond), an address (when one user responds) or a collision (if two or more users respond). A collision requires further queries to identify the active users in that set except for the case where the set contains only two users.

We describe this algorithm in terms of four sets of users, U, A(1), A(2) and K. As before U is the unknown set and K is the known set. Now there are two active sets A(1) and A(2). For i = 1, 2, A(i) is a set which is known to contain at least i active users. At any instant of time if $A(1) \neq \emptyset$, then $A(2) = \emptyset$ and vice versa. If A(2) is not empty, the algorithm queries a subset of it. If A(1) is not empty, however, the set queried is either a subset of A(1) or perhaps the union of A(1) with a subset of U. When a subset of A(2) is queried and a single address is received as response, the remaining users in A(2) form the set A(1). When a subset X of A(2) is queried and the response is a collision, the queried subset becomes the new set A(2) and the users in A(2) - X are combined with the users in U to form the new unknown set.

The complete algorithm is given below:

```
0.  U := S; K := ∅; A := ∅

1.  if U = ∅ then goto 4
              else query a subset X ⊆ U
                   if response is idle or a single address
                      then K := K + X;
                           U := U - X;
                           goto 1
                   if response is collision
                      then A(2) := X;
                              U := U - X;
                           If A(2) contains 2 users
                              then K := K + A(2);
                                     goto 1
                              else goto 2

2.  query a subset X ⊆ A(2)
           if response is idle
              then K := K + X;
                 A(2) := A(2) - X;
                 goto 2.
           if response is single address
              then K := K + X;
                 A(1) := A(2) - X;
                 If A(1) contains 1 user
                    then K := K + A(1);
                         goto 1
                    else goto 3
```

```
          if response is a collision
             then U := U + A(2) - X;
                  A(2) := X;
                  If A(2) contains 2 users
                     then K := K + A(2);
                          goto 1
                     else goto 2

3.  Query subset X where X ⊆ A(1) or X = A(1) + Y where Y ⊆ U
          if response is idle
             then K := K + X;
                  A(1) := A(1) - X;
                  If A(1) contains 1 user
                     then K := K + A(1);
                          goto 1
                     else goto 3
          if response is single address
             then K := K + X,
                  U := U - Y;
                  goto 1
          if response is collision
             then A(2) := X;
                     U := U - Y;
                     goto 2

4.  end
```

Recursion relations have been derived for the average number of queries assuming that the size of all subsets are chosen to result in this quantity being a minimum. The actual equations are not given here but the following table gives numerical results obtained from these equations. For comparison, corresponding results obtained using the tree algorithm of Gudjohnsen are given in parentheses.

Table II

Expected Number of Queries for Gudjohnsen (Capetanakis) Model

p	.1	.2	.3	.4	.5
N					
2	(1)1	(1)1	(1)1	(1)1	(1)1
3	1.038	1.14	1.31	1.51	1.75
4	(1.1)1.1	(1.33)1.33	(1.65)1.65	(1.99)1.99	(2.0)2.0
5	1.18	1.60	2.14	2.43	2.69
6	1.28	1.89	2.55	2.94	3.0
7	1.39	2.21	2.91	3.42	3.67
8	(1.53)1.52	(2.59)2.57	(3.30)3.3	(3.98)3.87	(4.0)4.0
9	1.66	2.90	3.75	4.37	4.67
10	1.81	3.17	4.16	4.84	5.0
11	1.98	3.46	4.55	5.30	5.67
12	2.14	3.75	4.94	5.80	6.0
13	2.31	4.06	5.39	6.27	6.67
14	2.5	4.39	5.80	6.73	7.0
15	2.69	4.74	6.19	7.22	7.67
16	(2.96)2.89	(5.18)5.02	(6.61)6.58	(7.97)7.69	(8.0)8.0

It can be seen from the above table that for the special case of p=1/2, the average number of queries is equal to N/2 if N is even and seems to be approaching $\left\lfloor \frac{N}{2} \right\rfloor + 2/3$ for N large and odd. Indeed this behavior has been verified analytically with the exact expression for all odd N being given as $\left\lfloor \frac{N}{2} \right\rfloor + 2/3 + (1/2)(1/4)^{\lfloor N/2 \rfloor - 1}$.

References

CAP 79 Capetanakis, J.I., "Tree Algorithms for Packet Broadcast Channels," IEEE Trans. on Information Theory, Vol. IT-25, pp. 505-515, 1979.

DOR 43 Dorfman, R., "The Detection of Defective Members of Large Populations," Ann. of Math. Stat., Vol. 14, pp. 436-440, 1943.

HAY 78 Hayes, J.H., "An Adaptive Technique for Local Distribution," IEEE Trans. on Communications, Vol. COM-26, pp. 1178-1186, 1978.

GUD 80 Gudjohnsen, E., D. Towsley and J.K. Wolf, "On Adaptive Polling Techniques for Computer Communication Networks," ICC 80 Conference Record, Vol. 1, pp. 13.3.1-13.3.5, Seattle, Washington, 1980.

SOB 59 Sobel, M. and P.A. Groll, "Group Testing to Eliminate Efficiently All Defectives in a Binomial Sample," Bell Sys. Tech. Journal Vol. 38, pp. 1178-1252, 1959.

STE 57 Sterrett, A., "On the Detection of Defective Members of Large Populations," Ann. Math. Stat., Vol. 28, pp. 1033-1036, 1957.

CONFLICT RESOLUTION PROTOCOLS FOR RANDOM MULTIPLE-ACCESS CHANNELS WITH BINARY FEEDBACK*

TOBY BERGER AND NADER MEHRAVARI
School of Electrical Engineering
Cornell University
Ithaca, N.Y. 14853

ABSTRACT

The problem of conflict resolution in a random multiple-access, time-slotted, packet-switched channel with binary feedback is considered. Conflict resolution algorithms (CRA) for two different types of binary feedback which informs the users only about whether or not there was a collision in the previous slot achieves a throughput of 0.442. The CRA proposed for the case in which the feedback informs the users about whether or not the previous slot was empty achieves a throughput of 0.279.

1. INTRODUCTION

Random multiple-access, time-slotted, packet-switched, collision-type channels with many bursty users and feedback have been investigated by a number of authors [1-9]. The main feature of these problems is to design and analyze efficient conflict resolution algorithms (CRA) for transmissions and retransmissions of packets. In [1-7] it is assumed that immediately after each slot a ternary feedback informs the users whether that slot was empty, or contained one packet (which was successfully transmitted), or contained a collision of two or more packets (which must be retransmitted at later times). This is referred to as 0,1,e feedback. The greatest known efficiency that can be achieved by CRAs employing 0,1,e ternary feedback is about 0.488 [4,6].

In this paper we consider the random multiple-accessing of a time-slotted, packet-switched, collision-type channel with "binary feedback". We consider two different types of binary feedback and introduce and analyze CRA's for each. The first one, referred to as "Conflict-No Conflict" feedback, is introduced and analyzed in Section 3. Section 4 contains the description and analysis of the second one which is referred to as "Something-Nothing" feedback. These types of binary feedback have some practical significance which we will discuss briefly later.

2. MODEL AND NOTATIONS

Consider a large population of remote stations (users) attempting to exchange equal-length information packets over a multi-access broadcast channel. The generation time of information packets by the users is modeled as a Poisson point process on the time interval $T = (a,b]$ with intensity $\lambda > o$. Therefore, we can associate with each packet the time it was generated. A CRA is defined as a protocol that the users follow to access the channel. Time is divided into slots numbered $k = 1,2,3,\ldots$. On the basis of past history of the slots, at the end of the kth slot the CRA designates a subset $\psi_{k+1} \subset T$. Each packet in ψ_{k+1} is then transmitted during the (k+1)st slot. A transmitted packet successfully reaches its destination only if no other packet is transmitted in the same slot. If more than one packet is transmitted in a slot, the packets "collide" and must be retransmitted in subsequent slots.

---

*This work was support in part by NSF grant ECS79-21976.

Let M denote the random number of packets generated in the time interval T = (a,b], and let τ be the random number of slots employed until all packets generated in T are successfully transmitted. The efficiency (throughput) of CRA is defined by $\eta = \frac{E[M]}{E[\tau]}$.

3. CONFLICT-NO CONFLICT (C-NC) FEEDBACK

It is assumed that at the end of kth slot each user learns the value of the feedback Z_k, where

$$Z_k = \begin{cases} NC & \text{if } \psi_k \text{ contained} \leqslant 1 \text{ arrival (no conflict)} \\ C & \text{if } \psi_k \text{ contained} \geqslant 2 \text{ arrivals (conflict)} \end{cases}$$

We propose a first-come first-served (FCFS) CRA deploying a C-NC feedback which is described with the aid of state diagram of Figure 1. By FCFS we mean that no message is ever successfully transmitted before another message with an earlier generation time.

A neighbor of interval I is defined to be an interval adjacent to I on the right. Let ℓ_k be the length of ψ_k. Let x, β, and α, where $0 \leqslant \beta, \alpha \leqslant 1$, be parameters to be optimized subsequently. The states in Figure 1 dictate the following actions.

State x — ψ_{k+1} = neighbor of ψ_k of length x

State β — ψ_{k+1} = the leftmost fraction β of ψ_k

State N_α — ψ_{k+1} = neighbor of ψ_k of length $\frac{(1-\beta)\alpha}{\beta}\ \ell_k$

State $N_{\overline{\alpha}}$ — ψ_{k+1} = neighbor of ψ_k of length $\frac{(1-\alpha)}{\alpha}\ \ell_k$.

The algorithm is initiated in state x by letting ψ_1 = (o,x]. The state transitions in Figure 1 are indicated by arrows labeled by our knowledge of feedback Z_k. By using the fact that the number of arrivals in non-overlapping Poisson process segments are independent, it is not difficult to show that the state x is a Markov state; the others are not.

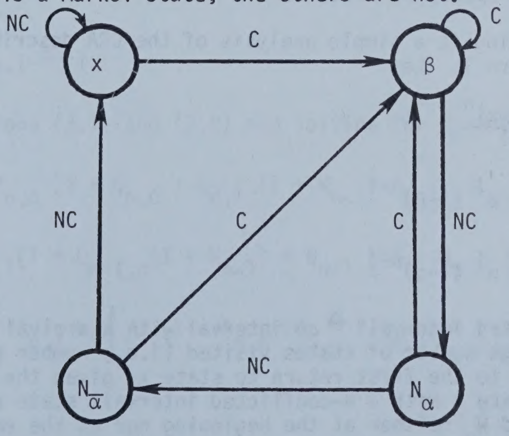


Figure 1. State diagram for CRA deploying C-NC feedback

The operation of this protocol is easily shown by an example. In Figure 2, the arrows along the time axis indicate times at which sources have generated messages for transmission. Assume all of the messages generated before time t have been successfully transmitted by using k slots.

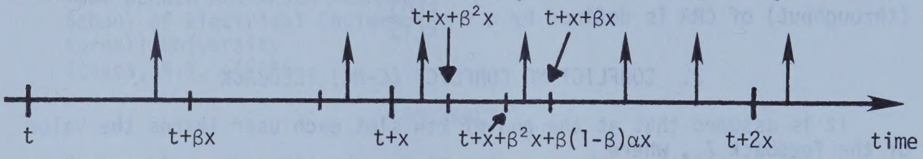


Figure 2. An example of arrival process

Therefore, the protocol resumes in state x by designating $\psi_{k+1} = (t, t+x]$. The feedback for each slot and the action of the protocol at each step for the example of Figure 2 is summarized in Table 1.

i	ψ_i	State	Z_i
k+1	$(t, t+x]$	x	C
k+2	$(t, t+\beta x]$	β	NC
k+3	$(t+\beta x, t+x\alpha(1-\beta)]$	N_α	NC
k+4	$(t+x\alpha(1-\beta), t+x]$	$N_{\overline{\alpha}}$	NC
k+5	$(t+x, t+2x]$	x	C
k+6	$(t+x, t+x+\beta x]$	β	C
k+7	$(t+x, t+x+\beta\beta x]$	β	NC
k+8	$(t+x+\beta\beta x, t+x+\beta\beta x+\beta(1-\beta)\alpha x]$	N_α	NC
k+9	$(t+x+\beta\beta x+\beta(1-\beta)\alpha x, t+x+\beta x]$	$N_{\overline{\alpha}}$	NC
k+10	$(t+x+\beta x, t+2x+\beta x]$	x	--

Table 1. ψ_i and Z_i for the example of Figure 2

Note that at the beginning of slot k+10, our knowledge concerning the number of arrivals in the interval $(t+x+\beta x, t+2x]$ has reverted to that of any unexplored portion of the arrival process of this length (i.e., Poisson distributed with parameter λ). The actions dictated by states N_α and $N_{\overline{\alpha}}$ may be viewed as having been selected in a manner that is a compromise between making the best use of the past history of the channel and assuring the Markovian property of state x.

The following is a simple analysis of the CRA described by the state diagram in Figure 1. Let

$$\Pi_n \triangleq \frac{e^{-\lambda x}(\lambda x)^n}{n!}$$

$$P_{n,i} \triangleq \binom{n}{i} \beta^i (1-\beta)^{n-i}$$

$$Q_{n,i} \triangleq \binom{n}{i} \alpha^i (1-\alpha)^{n-i}$$

"k-conflicted interval" $\triangleq$ an interval with k arrival in it

$W_n \triangleq$ average number of states visited (i.e., number of slots employed) prior to the first return to state x, given the protocol starts in state x with a n-conflicted interval; state x does not count toward W_n neither at the beginning nor at the end.

$C_n \triangleq \frac{1}{x}$ (average number of successful transmissions prior to the first return to state x, given the protocol starts in state x with a n-conflicted interval)

$W \triangleq$ average number of states visited (i.e., number of slots employed) between successive returns to state x

$C \triangleq$ average number of successful transmissions between successive returns to state x.

We have $W_o=0$, $W_1=0$, $C_o=1$, $C_1=1$, and the recurrence formulas for W_n, and C_n, where $n \geqslant 2$, are:

$$\begin{aligned} W_n = {} & 1 + P_{n,o}\left[1 + Q_{n,o}(1 + W_n) + Q_{n,1}(1 + W_{n-1}) + \sum_{i=2}^{n} Q_{n,i} W_i\right] \\ & + P_{n,1}\left[1 + Q_{n-1,o}(1 + W_{n-1}) + Q_{n-1,1}(1 + W_{n-2}) + \sum_{i=2}^{n} Q_{n,i} W_i\right] \\ & + \sum_{i=2}^{n} P_{n,i} W_i \end{aligned} \tag{3.1}$$

$$\begin{aligned} C_n = {} & P_{n,o}\Big[\beta + Q_{n,o}\left[\alpha(1-\beta) + (1-\alpha)(1-\beta)C_n\right] \\ & \qquad + Q_{n,1}\left[\alpha(1-\beta) + (1-\alpha)(1-\beta)C_{n-1}\right] \\ & \qquad + \sum_{i=2}^{n} Q_{n,i}\, \alpha(1-\beta)C_i\Big] \\ & + P_{n,1}\Big[\beta + Q_{n-1,o}\left[\alpha(1-\beta) + (1-\alpha)(1-\beta)C_{n-1}\right] \\ & \qquad + Q_{n-1,1}\left[\alpha(1-\beta) + (1-\alpha)(1-\beta)C_{n-2}\right] \\ & \qquad + \sum_{i=2}^{n-1} Q_{n-1,i}\, \alpha(1-\beta)C_i\Big] \\ & + \sum_{i=2}^{n} P_{n,i}\, \beta C_i \end{aligned} \tag{3.2}$$

Rewriting equations (3.1) and (3.2) and solving for W_n and C_n, we get

$$\begin{aligned} W_n = {} & \Bigg[1 + P_{n,o}\left[1 + Q_{n,o} + Q_{n,1}(1 + W_{n-1}) + \sum_{i=2}^{n-1} Q_{n,i} W_i\right] \\ & + P_{n,1}\left[1 + Q_{n-1,o}(1 + W_{n-1}) + Q_{n-1,1}(1 + W_{n-2}) + \sum_{i=2}^{n-1} Q_{n-1,i} W_i\right] \\ & + \sum_{i=2}^{n-1} P_{n,i} W_i\Bigg] \Big/ \left[1 - P_{n,o}(Q_{n,o} + Q_{n,n}) - P_{n,n}\right] \end{aligned} \tag{3.3}$$

$$\begin{aligned} C_n = \Bigg[& P_{n,o}[\beta + Q_{n,o}[\alpha(1-\beta)] + Q_{n,1}[\alpha(1-\beta) + (1-\alpha)(1-\beta) C_{n-1}] \\ & \quad + \sum_{i=2}^{n-1} Q_{n,1}\, \alpha(1-\beta) C_i] \\ & + P_{n,1} [\beta + Q_{n-1,o} [\alpha(1-\beta) + (1-\alpha)(1-\beta) C_{n-1}] \\ & \quad + Q_{n-1,1} [\alpha(1-\beta) + (1-\alpha)(1-\beta) C_{n-2}] \\ & \quad + \sum_{i=2}^{n-1} Q_{n-1,i}\, \alpha(1-\beta) C_i \\ & + \sum_{i=2}^{n-1} P_{n,i}\, \beta C_i \Bigg] \Big/ \ [1 - P_{n,o}(1-\beta)[Q_{n,o}(1-\alpha) + Q_{n,n}\alpha] - P_{n,n}\beta] \end{aligned} \tag{3.4}$$

Averaging over the Poisson random variable n, we get

$$W = 1 + \sum_{n=o}^{\infty} W_n \Pi_n \tag{3.5}$$

$$C = x \sum_{n=o}^{\infty} C_n \Pi_n \tag{3.6}$$

where W_n and C_n are given by equations (3.3) and (3.4). It follows that the efficiency of this CRA, as a functions of β, α, and x is given by $\eta_{C\text{-}NC} = C/W$ where C and W are given by equations (3.5) and (3.6). Optimization over β, α, and x shows that the maximum achievable throughput for the above CRA is 0.4422..., corresponding to $\beta = 0.435...$, $\alpha = 0.565...$, and $\lambda x = 1.255...$.

Recall that the greatest known efficiency that can be achieved by a CRA deploying 0,1,e feedback is about 0.488. Thus, in cases where the feedback is supplied to the users from a central facility, simplifying the feedback system of a random-access channel from 0,1,e type to C-NC reduced the throughput of the channel by only 10 percent.

4. SOMETHING-NOTHING (S-N) FEEDBACK

The second type of binary feedback is characteristic of public-key secure computer communications. Consider a random-access broadcast system where information packets are encoded by individual users using the public encryption key of the intended recipient. If, for example, user i is the only user who has transmitted a packet during the kth slot, then simply by listening to the channel (s)he will be aware of the fact that the packet has been successfully transmitted. Meanwhile, any user j ≠ i other than the intended recipient cannot distinguish between a single message encrypted for someone else and a collision between two or more messages. However, for the protocol to be executable in a distributed fashion, it is necessary to inform all of the users whenever a packet has been successfully transmitted. This is done by sacrificing one slot and re-enabling (see state R below) some portions of the arrival process which have already been examined by the protocol.

Therefore, assume that at the end of kth slot each user learns the value of the feedback y_k, where

$$y_k = \begin{cases} S & \text{if } \psi_k \text{ contained} \geqslant 1 \text{ arrival (something)} \\ N & \text{if } \psi_k \text{ conatined no arrivals (nothing)} \end{cases}$$

We propose a FCFS CRA deploying S-N feedback which is described with the aid of state diagram of Figure 3. The states in Figure 3 dictate the following actions.

State x — ψ_{k+1} = neighbor of ψ_k of length x

State β — ψ_{k+1} = the leftmost fraction β of ψ_k

State N_β — ψ_{k+1} = neighbor of ψ_k of length $(1 - \beta)\,\ell_k$

State $N_{\bar\beta}$ — ψ_{k+1} = neighbor of ψ_k of length $\dfrac{(1 - \beta)}{\beta}\,\ell_k$

State R — ψ_{k+1} = ψ_k with the exception that, if there was only one packet in the kth slot, the user who transmitted it and hence is aware that there was no collision, does not retransit the packet (s)he just successfully transmitted.

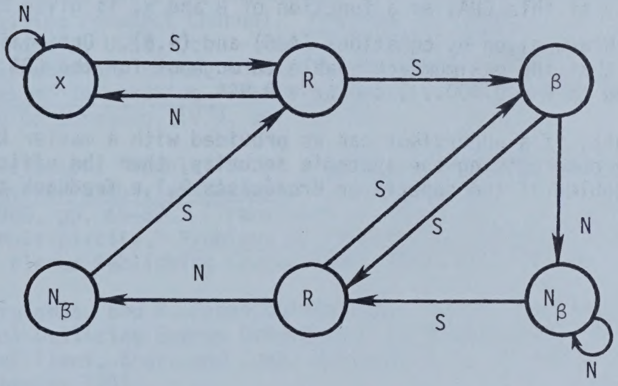


Figure 3. State diagram for the CRA deploying S-N feedback

The algorithm is initiated in state x by letting ψ_1 = (0,x]. The following is a simple analysis of the CRA described by the state diagrams in Figure 3. The definitions of Π_n, $P_{n,i}$, C_n, W_n, C , and W are the same as those used in section 3. We have $W_o = 0$, $C_o = 1$, and $C_1 = 1$. Note that here it takes two slots to process an interval containing one arrival. The first one is used to successfully transmit the single packet; the feedback from the second slot informs the rest of the users that the interval contained exactly one arrival. Hence, $W_1 = 1$. The recursive formulas for W_n and C_n, where $n \geqslant 2$, are

$$W_n = P_{n,o}\,(1 + W_n) + P_{n,1}\,(4 + W_{n-1}) + \sum_{i=2}^{n} P_{n,i}\,(2 + W_i) \tag{4.1}$$

$$C_n = P_{n,o}[\beta + (1 - \beta)C_n] + P_{n,1}[\beta + (1 - \beta)C_{n-1}] + \sum_{i=2}^{n} P_{n,i}\beta C_i \tag{4.2}$$

Rewriting equations (4.1) and (4.2) and solving for W_n and C_n, we get

$$W_n = [1 + P_{n,1}(3 + W_{n-1}) + P_{n,n} + \sum_{i=2}^{n-1} P_{n,i}(1 + W_i)]/(1 - P_{n,o} - P_{n,n}) \tag{4.3}$$

$$C_n = [\beta P_{n,o} + P_{n,1}]\beta + (1-\beta)C_{n-1}] + \sum_{i=2}^{n-1} P_{n,i}\beta C_i]/(1-\beta P_{n,n} - (1-\beta)P_{n,o}) \tag{4.4}$$

Averaging over n, we get

$$W = 1 + \sum_{n=o}^{\infty} W_n \Pi_n \tag{4.5}$$

$$C = x \sum_{n=o}^{\infty} C_n \Pi_n \tag{4.6}$$

where W_n and C_n are given by equations (4.3) and (4.4). It follows that the efficieny of this CRA, as a function of β and x, is given by $\eta_{S-N}=C/W$ where C and W are given by equations (4.5) and (4.6). Optimizing over β and x shows that the maximum achievable throughput for the CRA is 0.2794.., corresponding to $\beta = 0.400...$, and $\lambda x = 0.944...$.

Note that, if a supervisor can be provided with a master key without unacceptably compromising the system's security, then the efficiency can be nearly doubled if the supervisor broadcasts 0,1,e feedback to all the users.

REFERENCES

1. J. I. Capetanakis, "Tree Algorithm for Packet Broadcast Channels," IEEE Trans. on Information Theory, Vol. IT-25, No. 5, September 1979, pp. 505-515.

2. J. I. Capetanakis, "Generalized TDMA : The Multi-Accessing Protocol," IEEE Trans. on Communications, Vol. COM-27, No. 10, October 1979, pp. 1476-1484.

3. R. G. Gallager "Conflict Resolution in Random Access Broadcast Networks," Proc. of the AFOSR Workshop in Communications Theory and Applications, Provincetown, MA, 17-20, September 1978, pp. 74-76.

4. T. Berger, "The Poisson Multiple-Access Conflict Resolution Problem," Multi-User Communications, Edited by G. Longo, Springer Verlag, CISM Courses and Lecture Series, No. 265, 1981.

5. J. L. Massey, "Collision-Resolution Algorithms and Random-Access Communications," Multi-User Communications, Edited by G. Longo, Springer Verlag, CISM Courses and Lecture Series, No. 265, 1981.

6. J. Mosely, "An Efficient Contention Resolution Algorithm for Multiple Access Channels," Technical Report LIDS-TH-918, MIT, Cambridge, MA, June 1979.

7. B. S. Tsybakov and V. A. Mikhailov, "A Slotted Multiaccess Packet Broadcasting Feedback Channel," Problemy Peridachi Informatsii, Vol. 14, No. 4, Moscow, USSR, Oct. - Dec. 1978, pp. 32-59. (Translated as "Free Synchronors Packet Access in Broadcast Channel with Feedback," Problems of Information Transmission, Vol. 14, No. 4, Plenum Publishing Corporation, 1979, pp. 259-280.)

8. B. S. Tsybakov, "Resolution of a Conflict of Known Multiplicity," Problemy Peridachi Informatsii, Vol. 16, No. 2, Moscow, USSR, April - June 1980, pp. 69-82. (Translated as "Resolution of a Conflict with Known Multiplicity," Problems of Information Transmission, Vol. 16, No. 2, Plenum Publishing Corporation, 1980, pp. 134-144.

9. L. Georgiadis, and P. Papantoni-Katakos, "A Collision Resolution Protocol Utilizing Energy Detectors," Technical Report EECS-TR-81-4, Dept. of Elect. Engr. and Comp. Science, Univ. of Connecticut, Storrs, CT, February 1981.

DISTRIBUTED ASSIGNMENT OF LINKS IN MOBILE RADIO NETWORKS

D. J. BAKER
NRL
Washington, DC

A. EPHREMIDES
U. Of Md. & NRL
Washington, DC

J. WIESELTHIER
NRL
Washington, DC

Certain mobile radio users can be organized by means of recently derived distributed algorithms [1 - 3] into survivable networks with a resulting architecture of linked clusters of nodes. Such nets consist of clusters each of which is locally controlled by a uniquely determined clusterhead node and which communicate with each other via gateway nodes which are also unequivocally determined.

In this paper we address the open question of how to activate the potential links of the backbone network, namely those links that may connect clusterheads and gateways with each other. The activation of the links is a nontrivial problem due to the distributed nature of the information that concerns the identities of the resulting clusterhead and gateway nodes. We consider three methods for solving this problem and explore their advantages and disadvantages. Then we propose a fourth method which we consider perhaps to be preferable.

The first method relies on simulcast messages, the second one assumes a TDMA structure with frame length equal to the maximum number of neighbors a node may have, the third one calls for the formation of variable length TDMA frames with a procedure for solving inconsistent assignments and the fourth one utilizes the architecture-forming algorithm itself to pass on some additional information that produces the necessary data to allow for efficient variable length TDMA frame utilization. The last method does not have many of the shortcomings of the first three methods and furthermore it allows all nodes (not just the backbone nodes) to participate in the link activation process.

References

[1] D. J. Baker, A. Ephremides, "A Distributed Algorithm for Organizing Mobile Radio Telecommunication Networks", Proc. of 2nd Int'nl Conference on Distributed Computing, INRIA, April 1981.

[2] A. Ephremides, D. J. Baker, "An Alternative Algorithm for the Distributed Organization of Mobile Users into Connected Networks", Johns Hopkins Conference on Info. Sciences and Systems, March 1981.

[3] D. J. Baker, A. Ephremides, "The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm", IEEE Trans. on Comm., to appear.

ANALYSIS OF THE PACKET FORMATION PROCESS IN PACKET-SWITCHED NETWORKS*

J. S. MEDITCH
Department of Electrical Engineering
University of Washington
Seattle, Washington 98195

ABSTRACT

Two new queueing system models for the packet formation process in packet-switched telecommunication networks are developed, and their applications in process stability, performance analysis, and optimization studies are illustrated. The first, an M/M/1 queueing system characterization of the process, is a highly aggregated model which is useful for preliminary studies. The second, a marked extension of an earlier M/G/1 model, permits one to investigate stability, performance characteristics, and design of the packet formation process in terms of the details of processor architecture, and hardware and software implementations with processor structure and as many parameters as desired as variables. The two new models together with the earlier M/G/1 characterization span the spectrum of modeling complexity for the packet formation process from basic to advanced.

1. INTRODUCTION

One of the key processes associated with the operation of a packet-switched telecommunication network is that of subdividing incoming messages into packets for transmission through the network. At each destination, the packets are then reassembled to obtain the original messages. This message subdivision process, commonly termed the packet formation process, takes place at the switching nodes along with a variety of other processes such as numbering and sequencing, flow control, error control, routing, buffering, and packet reassembly into messages.

In [1], we formulated and studied an M/G/1 queueing system model for the packet formation process in which there were four process parameters: the average message arrival rate, the average message length, the maximum length of the text field in a packet, and the processor time to form a packet. All of these are physically meaningful and physically measurable variables, and the resulting model is useful for determining the stability limits and performance characteristics of the process. Our objective in this paper is to formulate and investigate two other models for the process which, in terms of detail and complexity, are on either side of the model in [1]. In the case of the simpler of the two models, our intent is to provide a means for quick and simple determination of process behavior using as few process parameters as are essential to be physically relevant. In the second model, we focus upon an extension of the model in [1] which, at the expense of increased detail, removes a limitation of the earlier model. Specifically, we remove the restriction that the processor time to form a packet be an independent parameter by allowing this time to be an arbitrary function of both the hardware and software implementations of the processor. Such a parameterization allows one to study the characteristics of the packet formation process as a function of processor architecture and implementation where the number of parameters may be quite large, e.g., in the tens if modest detail is required or on the order of one hundred or so

---

*Research supported in part by the U.S. Air Force Office of Scientific Research under AFOSR Grant 78-3546.

if significant detail is of interest.

The two models are M/M/1 and M/G/1 queueing systems, respectively. Aside from the two parameters which characterize the message arrival process and are identical for both models, the former model has three processor parameters, while the latter may involve up to 100 or more as remarked above.

Our presentation is as follows. We provide the common base for both models in Section 2, and then formulate and investigate the characteristics of the two models in Sections 3 and 4, respectively. We conclude in Section 5 with a discussion of our results and an indication of some possible issues for further investigation.

2. PACKET FORMATION PROCESS

The process with which we are dealing is depicted in Fig. 1 where the block called "Packet Formation" includes the queueing of arriving messages which are then serviced on a first-come/first-served basis to convert them into packets. We assume that the message arrival process is Poisson with average arrival rate λ messages/sec, and that the message lengths are independent and exponentially distributed with average length $1/\mu$ bits. We also assume that the arrival times and message lengths are independent of each other.

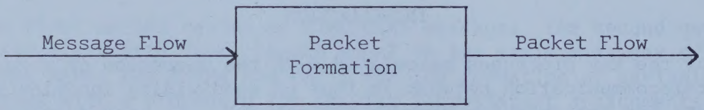


Figure 1. Packet formation process.

We consider the decomposition of messages into packets wherein the packet format is that shown in Fig. 2. The leading and trailing fields of each packet are assumed to be of fixed length, say M and N bits, respectively. These fields will typically be comprised of elements such as clock synchronization bits, control characters for text delimiting, source-destination addresses, protocol information, and hardware and software error checksums. The text field is taken to be variable up to a maximum of B

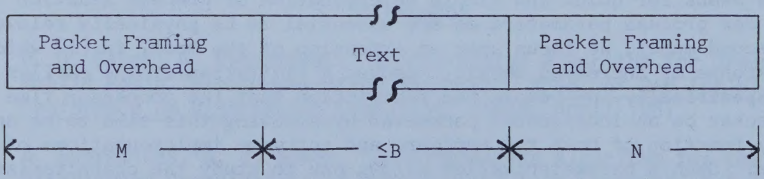


Figure 2. Packet format.

bits, the value of B being a parameter in both models. For a given value of B, a message of say length 3.21 B bits would be decomposed into three packets, each of length M+N+B bits, and one packet of length M+N+0.21B bits.

Finally, in both models, we let y denote the time to convert a message into packets, i.e., y is the processor service time per message, and we let $x \geq 0$ be the message length. The units on y and x are sec/message and bits/message, respectively.

3. M/M/1 MODEL

In this model, we take the service process to be defined by the relation

$$y = ax \tag{1}$$

where $a > 0$ is characterized by

$$a = b + \frac{c}{B} \tag{2}$$

In Eq. (2), the physical interpretation of the processor parameters is the following: b is the rate in sec/bit at which the message is read when it enters service, the ratio x/B specifies the number of packets into which the message is to be converted, and c is the rate in sec/packet at which the processor forms packets. The units on a in Eq. (1) are thus obviously sec/bit.

From Eq. (1) and the assumptions in Sec. 2 on the message arrival and message length processes, it follows that the packet formation model here is an M/M/1 queueing system for which the average service time is

$$\overline{y} = a\overline{x} = \frac{a}{\mu} \tag{3}$$

where the bar denotes the expected value and a is given by Eq. (2). Parenthetically, we note that the average number of packets per message is obviously

$$\overline{n} = \frac{\overline{x}}{B} = \frac{1}{\mu B}$$

For an M/M/1 queueing system, it is well known [2] that a necessary and sufficient condition for the system to be stable in the sense that the queue of messages waiting for service is finite for all time is that the average utilization $\rho = \lambda\overline{y}$ be strictly less than unity. From this, and Eqs. (2) and (3), we readily have the five-parameter stability requirement

$$b + \frac{c}{B} < \frac{\mu}{\lambda} \tag{4}$$

Specification of any four of the parameters then clearly dictates a bound on the other. A case of some interest is that in which the message process (λ,μ) and the processor rates (b,c) are given. The relation in Eq. (4) then leads to

$$B > \frac{c\lambda}{\mu - b\lambda} \tag{5}$$

as a lower bound on packet size. Since $B > 0$ is a requirement for the packet formation process to make physical sense, we obtain also the condition $\mu > b\lambda$.

Further, for an M/M/1 queueing system, we have that the average waiting time (average time in queue per message) and the average system time (average waiting time plus average service time per message) are given [2] by

$$W = \frac{\rho\overline{y}}{1-\rho}$$

and

$$T = \frac{\overline{y}}{1-\rho}$$

respectively. From Eqs. (2) and (3), and the definition of ρ, we get

$$T = \frac{bB+c}{\mu B-(bB+c)\lambda} \tag{6}$$

for our system where the units on T are sec/message to convert to packets.

It is obviously a simple matter to utilize Eq. (6) to conduct tradeoff studies of the average system time as a function of the five parameters or certain ratios thereof on the right-hand side of the equation. The same also applies to the corresponding expression for W. We focus here instead on the examination of T as a function of B in an effort to obtain bounds on the maximum text length of a packet which will guarantee a specified level of average system delay.

We observe first from Eq. (6) that the theoretical minimum of T as a function of B occurs when we let $B \to \infty$. In this case, we get

$$T_{min} = \frac{b}{\mu-b\lambda} \tag{7}$$

provided that $\mu > b\lambda$ as was required in Eq. (5). This, of course, corresponds to the implementation in which each message is converted into a single packet. More practically, however, we note that if we, for example, select B such that $bB = 100c$, we have from Eq. (6) that

$$T = \frac{1.01b}{\mu-1.01\lambda}$$

which very closely approximates T_{min} provided that $\mu > 1.01\lambda$.

To generalize, we let

$$B \triangleq \frac{(\alpha+1)c\lambda}{\mu-b\lambda} \tag{8}$$

for fixed (λ,μ,b,c) with α a parameter. From Eq. (5), we have the requirements $\mu > b\lambda$ and $\alpha > 0$. Substitution of Eq. (8) into Eq. (6) and utilization of Eq. (7) gives

$$T(\alpha) = \frac{b}{\mu-b\lambda} + \frac{\mu}{\alpha\lambda(\mu-b\lambda)}$$
$$= T_{min} + T^*(\alpha) \tag{9}$$

where

$$T^*(\alpha) \triangleq \frac{\mu}{\alpha\lambda(\mu-b\lambda)} \tag{10}$$

Equation (9) clearly reveals the "irremovable" and "adjustable" components of the average system delay, viz., T_{min} and $T^*(\alpha)$, respectively.

As an illustration of the use of Eq. (9), let us suppose that the acceptable level of average system delay, say relative to other delays in the network, is in the range $0.1\ T_{min} \leq T^*(\alpha) \leq 0.2\ T_{min}$. From Eqs. (7), (10), and (8), we get

$$\frac{c(b\lambda+5\mu)}{b(\mu-b\lambda)} \leq B \leq \frac{c(b\lambda+10\mu)}{b(\mu-b\lambda)}$$

as the required range on B.

4. M/G/1 MODEL

The service process here is significantly more detailed and complete than that above and the one in [1]. In particular, we assume that the time required by the packet formation process to compose one packet, including packet framing and text, is given by $\tau = \tau(B,\pi)$ sec/packet where π is the vector of processor architecture, hardware, and software parameters. The exact form of $\tau(\cdot,\cdot)$ is structure dependent. It is taken to be general in the development below, but a specific form will be chosen to illustrate applications of our results. The distinction to be made here relative to the model in [1] is that τ was assumed to be independent of B and π in the latter.

The model for the processor service time is taken to be

$$y = n\ \tau(B,\pi) \tag{11}$$

where $n = k$ if $(k-1)B < x \leq kB$, $k = 1, 2, \cdots$. Defining $p_k = P[y=k\tau(B,\pi)]$ where $k = 0, 1, \cdots$ and $P[\cdot]$ denotes the probability, and utilizing the fact that the message lengths are independent and exponentially distributed with mean $1/\mu$, it is a direct matter to show that

$$p_k = \begin{cases} 0 & \text{for } k = 0 \\ e^{-(k-1)\mu B}(1-e^{-\mu B}) & \text{for } k = 1, 2, \cdots . \end{cases} \tag{12}$$

Since the message arrival process is Poisson, it follows that the packet formation process is an M/G/1 queueing system in which the general service process is specified by Eqs. (11) and (12). This permits us to employ a number of known queueing theoretic results including the Pollaczek-Khinchin (P-K) mean value formula [2] in order to characterize the properties of this process.

We consider first the average service time

$$\overline{y} = \tau(B,\pi) \sum_{k=0}^{\infty} kp_k \tag{13}$$

for converting messages into packets. Substituting into Eq. (13) from Eq. (12), we find after some algebra that

$$\overline{y} = \frac{\tau(B,\pi)}{1 - e^{-\mu B}} \tag{14}$$

where the units on $\overline{y}$ are sec/message. From Eqs. (11) and (14), it is obvious that the average number of packets per message is

$$\overline{n} = \frac{1}{1 - e^{-\mu B}} \tag{15}$$

In order to utilize the P-K mean value formula, we shall also need the second moment of y which is defined by

$$\overline{y^2} = \tau^2(B,\pi) \sum_{k=0}^{\infty} k^2 p_k$$

Using Eq. (12) again, it is relatively easy to show that

$$\overline{y^2} = \frac{(1 + e^{-\mu B})\ \tau^2(B,\pi)}{(1 - e^{-\mu B})^2} \tag{16}$$

We turn now to the matter of average utilization and the issue of the stability of the packet formation process. The former is seen to be

$$\rho = \frac{\lambda\pi(B,\pi)}{1 - e^{-\mu B}} \tag{17}$$

If it is specified that the packet formation process is to operate at a certain utilization, say 0.80, Eq. (17) may be used to define the parameter space (λ,μ,B,π) over which this utilization can be achieved.

Invoking the stability requirement $\rho < 1$, we get

$$\frac{\lambda\pi(B,\pi)}{1 - e^{-\mu B}} < 1 \tag{18}$$

which may be used to define parametric stability boundaries. As a simple illustration, we consider the case where $\tau = a + Bc$. Here, a and c are processor parameters with $a > 0$ being the assumed fixed time which it takes the processor to frame a packet and $c > 0$ is assumed to be the rate (in sec/bit) at which the processor forms the text field. The value of τ then represents an upper bound on packet formation time since many packets will be of length less than B. Taking $\lambda = 100$ messages/sec, $1/\mu = 1000$ bits, and B = 800 bits, we have from Eq. (18) the condition $a + 800\,c < 5.51$ mesc. Any choice of a and c for which this relation holds then guarantees the stability of the packet formation process.

We next provide the delay analysis equation for the process and comment on some of its uses. The average time that a message spends in the process, i.e., its time in queue and service, is $T = \overline{y} + W$ where W is the queueing time and is given by the P-K relation [2]

$$W = \frac{\lambda \overline{y^2}}{2(1-\rho)} \tag{19}$$

It then follows from Eqs. (14), (16), (17), and (19) that

$$T = \frac{\tau(B,\pi)[2-\lambda\tau(B,\pi)]}{2[1-e^{-\mu B}-\lambda\tau(B,\pi)]} \tag{20}$$

Equation (20) obviously has numerous applications in parametric studies of the packet formation process, perhaps the simplest of these being the delay vs. offered traffic (T vs. λ) characteristic for fixed $\tau(\cdot,\cdot)$, μ, B, and π. Alternately, a more important study might be the one in which $\tau(\cdot,\cdot)$, μ, B, and λ are fixed, and it is desired to determine processor parameter vectors π such that a specified delay T is achieved. For this purpose, one can derive from Eq. (20) the relation

$$\tau(B,\pi) = \frac{(1+\lambda T) - \sqrt{1+2\lambda T e^{-\mu B}+\lambda^2 T^2}}{\lambda} \tag{21}$$

where the right-hand side is a known constant. The design problem then becomes that of finding one or more physically meaningful parameter vectors π which satisfy Eq. (21). If none exist, then it may be necessary to relax the delay requirement and/or seek a different processor architecture.

To illustrate the use of Eq. (20) for purposes of optimizing system delay, we consider a case where λ = 2000 messages/sec, $1/\mu$ = 1000 bits, and $\tau(B,\pi) = a + Bc$ where a = 0.30 msec and c = 1.00 msec/bit. For these parameter values, Eq. (20) becomes

$$T = \frac{(10^{-3}B+0.3)[1-(B+300)\times 10^{-6}]}{1 - e^{-.001B} - 2(B+300)\times 10^{-6}} \quad \text{msec}$$

This is a convex function of B which ranges in behavior from T = 4.24 msec for B = 100 bits to T = 5.36 msec for B = 5000 bits, and has a global minimum of T = 2.00 msec when B = 690 bits. The minimum is, however, relatively flat in the range $500 \le B \le 1000$ bits wherein T starts at 2.04 msec, decreases monotonically to the above global minimum, and then increases monotonically to 2.06 msec. This example serves to make clear the existence of optimal choices of the packet formation process parameters. In this case, the optimum turned out to be very flat thereby providing options in selecting B by bringing other design considerations such as buffer space into play.

5. CONCLUSION

In this paper, we have presented two new queueing system models for the packet formation process in packet-switched telecommunication networks, and have illustrated potential applications of these models for both performance analysis and optimization. In both models, the message arrival process was assumed to be Poisson with average rate λ and the message lengths were taken to be exponentially distributed with mean $1/\mu$. The first model, an M/M/1 queueing system characterization of the packet formation process, is considerably simpler than the M/G/1 model in [1]. Although the former involves the three processor parameters b, c, and B which were described in connection with Eq. (2), it is fundamentally characterized by the single, aggregate parameter a introduced in Eq. (1).

Using the specific interpretation of a which was prescribed by Eq. (2), the basic relations which we formulated for the M/M/1 model were Eq. (4) as a general stability condition, Eq. (5) as a lower bound on the maximum text field size B for stability, Eq. (6) as the basic system delay relationship in terms of the parameter set (λ,μ,b,c,B), Eq. (7) which specifies the theoretical minimum for the average system time, and Eq. (9) which reveals the intrinsic structure of Eq. (6) and facilitates performance analysis and design.

The M/G/1 model in Section 4 is a detailed extension of the one in [1], making it possible to investigate the stability and performance of the packet formation process as a function of the processor architecture and its hardware and software implementations. The basic performance analysis and optimization relations which were developed and illustrated were Eq. (14) for the average service time to convert a message into packets, Eq. (17) for the average utilization of the processor, Eq. (18) as the necessary and sufficent condition for stability, and Eq. (20), the average system delay relation, which can be used for either tradeoff studies or minimization of system delay.

In practice, the delay introduced by the packet formation process is typically small relative to that due to the communications subnetwork. For example, in ARPANET, TELENET, and TYMNET, the communications delay is on the order of 200 or so msec. The value, then, of the models and their associated performance relations which we have presented here and in [1] perhaps resides more in assuring that nodal processing resources are efficiently utilized for the packet formation process than it does in necessarily "fine tuning" the process parameters to reduce process delay from say 4.5 msec to 4.1 msec. On the other hand, it is clear from the performance relations that careless choices of process parameters can not only lead to poor performance, but worse yet to unstable behavior.

Certain issues associated with the packet formation process which may merit investigation in the future include characterization of the output process, viz., the packet flow in Fig. 1, the modeling of the process for messages with priorities, the possibility of parallel processing, i.e., multiple servers, and the effects of errors in packet formation.

REFERENCES

1. J. S. Meditch, "Modeling and performance analysis of the packet formation process in packet-switched telecommunications," *Proc. 18th Allerton Conf. on Comm., Contr., and Comp.*, Univ. of Ill., Urbana, Ill., Oct. 1980, pp. 333-338.

2. L. Kleinrock, *Queueing Systems: Vol. II, Computer Applications*. New York: Wiley-Interscience, 1976.

DISTRIBUTED MANAGEMENT SYSTEM OF CRIPTOGRAPHIC KEYS IN COMMUNICATION NETWORKS

E. OKAMOTO, K. NAKAMURA, Y. SHIMIZU and Y. TESHIGAWARA
Nippon Electric Co., Ltd.
1-1 Miyazaki, Yonchome, Takatsu-ku, Kawasaki-city
Kanagawa Prefecture 213, Japan

Y. SATO
Toho University
2-1, Mitsuyama, Nichome, Funabashi-city
Chiba Prefecture 274, Japan

ABSTRACT

This paper proposes a "distributed" cryptographic key management system for broadcast communication networks which do not allow improper message alternations. Total security of the network equipped with the proposed key management system is not damaged, even if the security of a terminal is exposed. That is, the damage is localized in those networks. Furthermore, the damaged network recovers its normal status without difficulty.

1. INTRODUCTION

There is increasing growth in the number of local computer networks in use and in the kind of distributed computing applications available on these networks. This increase has created necessity of data integrity and interest in cryptography. Fig.1 illustrates a local network where messages are open to tapping by an eavesdropper.

The only general approach to sending data safely over insecure media is to use encryption techniques. Data among the terminals equipped with Encryptor/ Decryptors is secure (Fig.2). In modern cryptology, encryption algorithms are public. Even if steps are taken to conceal the algorithms, revelation is simply a matter of time. Overall security of the system is dependent on the secrecy of cryptographic keys used in the system. The use of the same key over a long period increases the possibility of the key revelation[1]. Therefore the key must be changed occasionally.

Most cryptosystems centralize key management for key generation, distribution and storage on a specific terminal (Fig.3). They have a grave defect that the exposure of the key management center to an unauthorized person causes the damage of the total network security. Decentralization of the key management to each terminal, if it is possible, decreases the damage and localizes its influence.

This paper proposes a distributed cryptographic key management system (hereafter, abbreviated as DKMS) for broadcast communication networks which do not allow improper message alternations. One example of such networks is considered to be Ethernet[2].

Fig.4 illustrates the local computer network with the proposed distributed key management facility. The distributed key management facility located in any one terminal changes the key between itself and other terminals, in cooperation with the distributed key management facilities located in the other terminals. In this network, in spite of the exposure of the cryptosystem at a terminal, the communication between other terminals is secure. Furthermore, the locally damaged network can recover its normal status without difficulty by the execution of the key-change procedure between the exposed terminal and the other terminals.

2. PRINCIPLES OF THE DKMS

The principal functions of the DKMS are key change including key distribution and terminal authentication.

2.1 Key-change scheme

In the DKMS, a key between any two termmals is changed to a new key by only the two terminals. Suppose they be T1, T2, and T1 take the initiative in changing the key. There are four schemes for key change method as shown in Figure 5. In the Figure, E_{KM} is an encrypting transformation used for key distribution. The key KM of E_{KM} is called a terminal master key. E_{KM} is another transformation different from the transformation of an Encryptor in Figure 4 and must be commutative[3];

$$E_{KMi} E_{KMj}(x) = E_{KMj} E_{KMi}(x)$$

where x is any message and KMi is the master key of terminal Ti. D_{KM} is a decrypting transformation corresponding to E_{KM}. RN is a random code generated by T1. The underlined codes are new keys for data (hereafter, called as data-keys).

In the case of the scheme (c) in Figure 5, the terminal T1 generates a random code RN and sends it to the other terminal T2. The terminal T2 encrypts RN to $E_{KM2}(RN)$ and sends it back to the terminal T1. The terminal T1 encrypts $E_{KM2}(RN)$ to $E_{KM1} E_{KM2}(RN)$ and resends it to T2. The terminal T2 decrypts it to $D_{KM2} E_{KM1} E_{KM2}(RN)$ which equals $E_{KM1}(RN)$, since E_{KM} is commutative. On the other hand, the terminal T1 is able to obtain the same code $E_{KM1}(RN)$ by encrypting RN on its own master key KM1. The common code $E_{KM1}(RN)$ is used as a new data key for encryption between T1 and T2. In the case of the scheme (a), a new data key is $E_{KM1} E_{KM2}(RN)$. This sheme (a) is the same as the public key distribution scheme by Diffie and Hellman[4].

Among (a), (b), (c) and (d), two schemes (b) and (c) are possible to be applied to multi-destination networks, since the data-key RN or $E_{KM1}(RN)$ depends only on the terminal T1's information. In the scheme (c), T1 need not decryption process, where as in the scheme (b) both T1 and T2 need decryption process. This paper deals with the scheme (c) principally.

It is almost impossible to obtain the code $E_{KM1}(RN)$, knowing codes RN, $E_{KM2}(RN)$ and $E_{KM1} E_{KM2}(RN)$, if the encryption transformation E_{KM} endures the so-called known plain-text attack. One of the examples which are commutative and stand known plain-text attack is

$$E_{KM}(x) = x^{KM} \pmod p$$

where p is a large prime number. Even if someone knows the number x, p and the cipher $x^{KM} \pmod p$, he could not obtain KM, because the example is known as one of the oneway functions.

It is possible to construct the crypptosystems which treat KMi as a random code and RN as a fixed code in Figure 5 like the public key distribution scheme[4]. The variant of the key-change scheme (c) in which KMi is treated as a random code and RN as a fixed code is less convenient to use, however, since each cryptosystem in terminals must remember random number KMi's. Each KMi is generated corresponding to every terminal.

2.2 Terminal authentication

The proposed key change scheme needs terminal authentication, since if another terminal disguises itself as the terminal T2 and practises the key-change scheme of T2's side using its own master key KM3, then the impostor terminal obtains a new data key $E_{KM1}(RN)$. To defend the system from impostors, the DKMS adopts the following rule.

Rule: every terminal must respond to a key-change request.
It is assumed that the network does not allow illegal message alternations. Therefore the terminal which has sent a key-change request necessarily receive a response. The plural responses indicates the existence of an impostor terminal. The terminal T1 could not, however, determine which response is right one. The terminal T1 cancels the key-change procedure on the way and retries it later. In the case of constant disturbance by an unknown terminal, T1 gives an alarm.

Authentications for such as time, message, etc. other than terminal authentication are also necessary. These are established by the same procedure of SNA[5] after key change scheme.

2.3 Key-change procedure

The key-change procedure is constructed using the key-change scheme (c) in Figure 5 and the rule for terminal authentication. Fig.6 indicates the flow-chart of one of the examples of the key-change procedure. In (a) of Fig.6 the terminal T1 taking the initiative in changing the data-key generates a random number RN and sends it to the terminal T2 together with a sequential number 1. A sequential number is used as a further means to detect an impostor terminal. At the same time T1 sets the number 1 to c, a pointer of state, and enters a waiting state. In (b) of Fig.6, both terminals T1 and T2 are waiting the messages such as "RN, 1" on key-change procedure. One of them receives "RX, sequential number". The terminal checks whether the sequential number equals c+1 or not. When it does not equal c + 1, there is an impostor terminal. If the sequential number equals c + 1, the terminal proceeds to the next step according to the sequential number. When the received sequential number equals 1, the terminal, in this case T2, encrypts the received code RX and sends it back with a sequential number 2 to the other terminal T1. The terminal T2 sets 2 to c and enters a waiting state. If the received sequential number is 2, the terminal, in this case T1, encrypts RX and sends it back with a sequential number 3 to T2. Moreover T1 encrypts RN to $E_{KM1}(RN)$, used as the new deta-key. The terminal T1 sets the sequential number 3 to c and enters a waiting state. If the received sequential number is 3, the terminal, in this case T2, decrypts RX to $D_{KM2}(RX)$ equal to $E_{KM1}(RN)$. The terminal 2 sets 0 to c. The code $E_{KM1}(RN)$ is used as the new data key between T1 and T2. Obeying the key-change procedure, the terminal T1 is able to check the existence of an impostor terminal with the aid of the sequential number and the pointer of state c. The messages on the key change procedure are also encrypted under the Encryptor/Decryptors with the old data key as a precaution.

In the case that the proper terminal is inactive and the data-key is exposed, the network security could be damaged by an impostor terminal, since the true terminal do not obey the rule for the terminal authentication. In this case, however, the network recovers its status by the activation of the right terminal and the execution of the key-change procedure.

3. THE DKMS STRUCTURE

The proposed DKMS equipped with functions of key-change and terminal authentication is situated in the network as shown in Fig.4. Fig.7 illustrates the situation of the DKMS with respect to the network architecture OSI[6] and the correspondence with Ethernet as an example. The cryptosystem is just over the network layer which offers data transmission function at network level.

Block diagram of cryptosystem is shown by Fig. 8. The cryptosystem is composed of Data Encryptor/Decryptor, Distributed Key Management Facility (DKMF) and Data Key Table. Data is encrypted or decrypted by the Data Encryptor/Decryptor using a data-key stored in the Data Key Table. Source addresses and destination addresses are not encrypted. Only data is encrypted or decrypted. At the time of key-change, the Selector connects between the DKMF and the Data Encryptor/Decryptor. The DKMF executes the key-change procedure, and as the result, a new data-key is distributed.

4. CONCLUSION

We proposed a distributed key management system to accomplish decentralization of key management in broadcast communication networks. The proposed DKMS has the following several features.

1) Network secrity is decentralized. Even if the cryptosystem of one terminal is exposed, the damage does not spread throughout the network.

2) Terminals need not send their master key to any other terminals.

3) Terminals can change their master keys arbitrarily.

4) Any one terminal can detect the existence of an impostor terminal within the network.

5) Even if the cryptosystem of a terminal is damaged, it can recover its status easily by the use of the key-change procedure.

6) It is easy to attach a new terminal with the distributed key management facility.

This paper describes only the outline of the distributed key management system. One must specify details for actual systems at a later date.

ACKNOWLEDGMENT

The authors would like to thank Mr. Y. Kato, Mr. T. Yagi, Mr. T. Ishiguro, Mr. Y. Matsuo and Mr. F. Akashi for their encouragements.

REFERENCE

1 C. E. Shannon, "Communication Theory of Secrecy Systems", BSTJ, Vol.28, pp.656-715, 1949
2 R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks", Comm. ACM, Vol.19, No.7, pp.395-404, 1976
3 G. J. Simmons, "Symmetric and Asymmetric Encryption", Cornputing Surveys, Vol.11, No.4, pp.305-330, 1979
4 W. Diffie and M. E. Hellman, "New Directions in Cryptography," IEEE Trans. Inf. Theory, Vol.IT-22, No.6, pp.644-654, 1976
5 R. E. Lennon, "Cryptography Architecture for Information Security", IBM Syst. J., Vol.17, No.2, pp.138-150, 1978
6 ISO/TC97/SC16 N34, International Standard Organization, Provisional Model of Open System Archtecture.

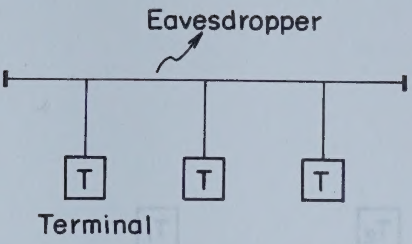

Eavesdropper
T
T
T
Terminal

Fig. 1 Local network and eavesdropper

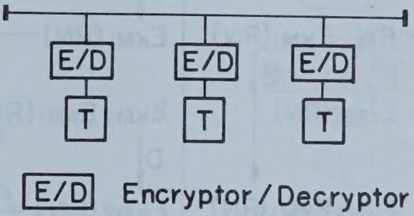

E/D
E/D
E/D
T
T
T
E/D Encryptor / Decryptor

Fig. 2 Local network with cryptography

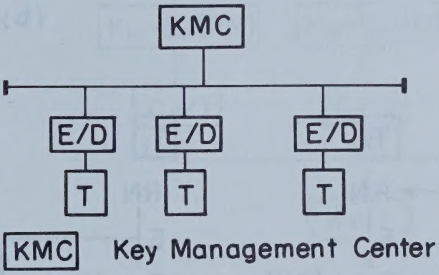

KMC
E/D
E/D
E/D
T
T
T
KMC Key Management Center

Fig. 3 Centralized key management

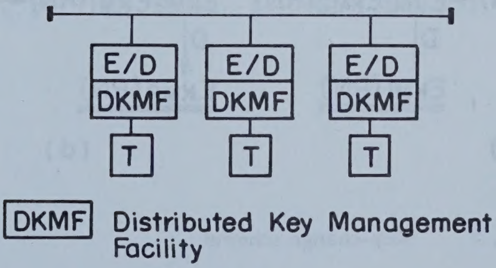

E/D
DKMF
E/D
DKMF
E/D
DKMF
T
T
T
DKMF Distributed Key Management Facility

Fig. 4 Distributed key management

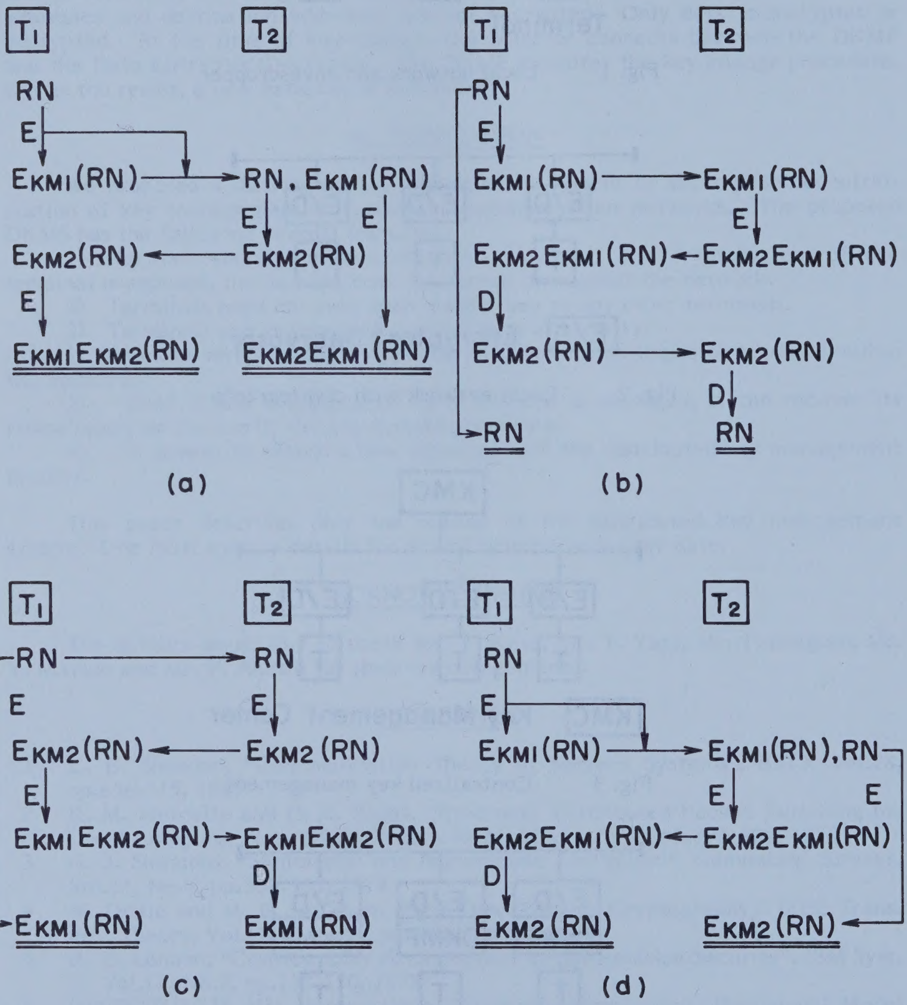

T1
T2
RN
E
EKM1(RN)
RN, EKM1(RN)
EKM2(RN)
EKM1EKM2(RN)
EKM2EKM1(RN)
(a)
EKM2EKM1(RN)
D
(b)
(c)
(d)

Fig. 5 Key-change scheme

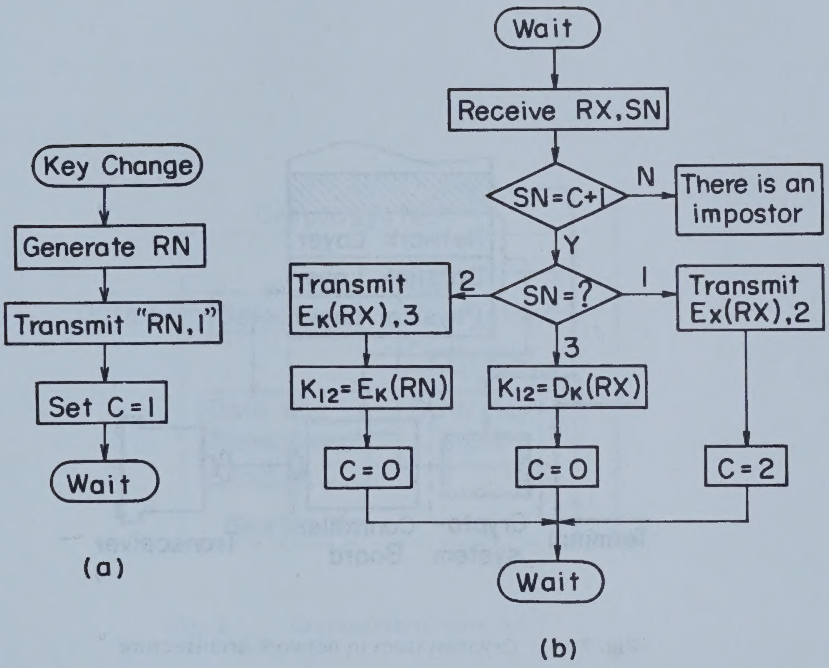

Key Change
Generate RN
Transmit "RN,1"
Set C=1
Wait
(a)
Wait
Receive RX,SN
SN=C+1
N
There is an impostor
Y
SN=?
2
1
3
Transmit E_K(RX),3
Transmit E_X(RX),2
K12=E_K(RN)
K12=D_K(RX)
C=0
C=0
C=2
Wait
(b)

Fig. 6 Flow chart of key-change procedure

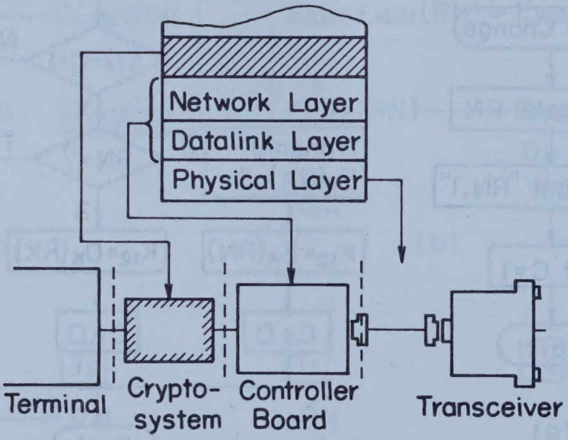

Network Layer
Datalink Layer
Physical Layer
Terminal
Crypto-system
Controller Board
Transceiver

Fig. 7 Cryptosystem in network architecture

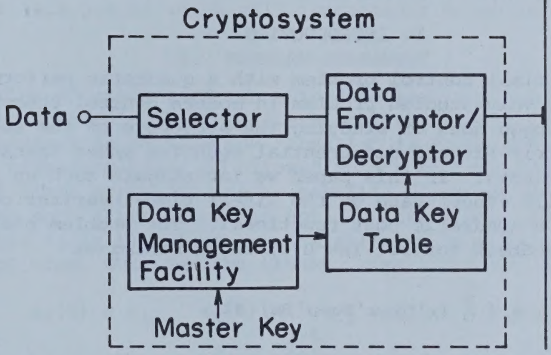

Cryptosystem
Data
Selector
Data Encryptor/ Decryptor
Data Key Management Facility
Data Key Table
Master Key

Fig. 8 Cryptosystem structure

THE RICCATI EQUATION, IMPRIMITIVE ACTIONS AND SYMPLECTIC FORMS

MARIO R. GARZIA
FPGD Information Systems Dept., Babcock & Wilcox Co., Barberton, Ohio 44203
KENNETH A. LOPARO+
Department of Systems Engineering, Case Institute of Technology, Cleveland, Ohio 44106
CLYDE F. MARTIN*+
Department of Mathematics, Case Institute of Technology, Cleveland, Ohio 44106

ABSTRACT

This paper looks at the structure of the solution of a matrix Riccati differential equation under a predefined group of transformations. The group of transformations used is an expanded form of the feedback group. It is shown that this group of transformations is a subgroup of the symplectic group. The orbits of the Riccati differential equation under the action of this group are studied and it is seen how these techniques apply to a decentralized optimal control problem.

I. INTRODUCTION

The linear optimal control problem with a quadratic performance criteria is an extensively studied problem in modern control theory. To date not much work has been done on studying the structure of the solutions of its associated matrix Riccati differential equation under transformation on its matrix coefficients. In this paper we investigate such an approach.

This study will concentrate on the linear time-invariant optimal control problem with a quadratic cost functional. The problem statement is: Find the linear feedback control law u(t) which minimizes

$$J = \int_0^T (x'Qx+x'Su+u'Ru)\, dt \tag{1a}$$

subject to

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1b}$$

where $Q \geq 0$ and A are nxn dimensional matrices, S and B are nxm and R is an mxm positive definite matrix. As is well known, the solution to (1) is related to the solution of the matrix Riccati differential equation

$$\dot{P}(t) = -A'P(t) - P(t)A + (P(t)B+S)R^{-1}(P(t)B+S)' - Q \tag{2}$$

This paper will be concerned with studying the solutions of (2) under the action of a group of transformations acting on the coefficient matrices.

The particular solution structure sought will be that for which the solution P(t) to (2) takes the block diagonal form

$$P = \begin{bmatrix} P_1 & & O \\ & \ddots & \\ O & & P_k \end{bmatrix} \tag{3}$$

+Supported in part by NASA Grants #2384 and NAG2-82; and ONR Contract #N000 14-80C-0199.
*Also in part by DOE Contract #DE-AC01-80RA5256.

for some integer $k \geq 0$, and the P_i of dimension $n_i x n_i$. This type of solution is of great importance. For instance, in the area of decentralized control. In this case, a solution of the form (3) implies optimal local feedback laws. Furthermore, using this approach, there are no approximation errors as with the classical techniques [7]. The approach used to carry out these parameter variations is also of theoretical importance since it deepens our understanding of the flow of the matrix Riccati differential equation.

In order to study the flow of (2) under parameter transformations, it is first necessary to define the "allowed" variations. The transformations used will be a subgroup of what in [1,2] was termed the Riccati group.

Section II contains a complete statement of the optimal decentralized control problem which will be used as motivation for the work which follows. In section III, the Riccati group and its actions on the Riccati differential equation will be discussed. Following [1], this action will be defined on the 5-tuple (A,B,Q,R,S) which along with an initial condition P_o completely defines the equation (2). In section IV it will be shown that the actions of the Riccati group on the associated Hamiltonian form a subgroup of the symplectic group. Once this is done, we will study the stabilizer of the group action. Necessary conditions for the stabilizer to have maximal dimension under this action will also be given. Section V contains the concluding remarks giving necessary conditions on the coefficients of (A,B,Q,R,S) which will guarantee a solution of the form (3).

II. PROBLEM STATEMENT

A problem of great importance in decentralized control is to determine when it is possible to apply feedback control laws "locally". What is equally important is to know how "locally" is to be defined. That is, when a group of transformations is allowed to act on the system, not every combination of state variables may be possible since the coefficient matrices representing such a structure may not lie within the orbit of the given system. This is an important consideration to keep in mind from a practical point of view. Consider the linear, time-invariant optimal control problem

$$x_i(t) = A_{ii} \quad x_i(t) + \sum_{\substack{j=1 \\ j \neq i}}^{k} A_{ij} \, x_j(t) + \sum_{l=1}^{p} B_{il} u_j \tag{4a}$$

$$x(o) = x_o \qquad\qquad i=1,\ldots,k$$

with quadratic cost functional

$$J = \int_0^T \quad (x'Qx + u'Ru) \; dt \tag{4b}$$

where x_i is the n_i-dimensional state vector, u_j is the m_j-dimensional control vector. The matrices $R>0$ and $Q \geq 0$ are mxm, and nxn symmetric matrices, respectively. A_{ij} is of dimension $n_i x n_j$ and B_{il} is of dimension $n_i x m_l$, with

$$\sum_{i=1}^{k} n_i = n \quad \text{and} \quad \sum_{i=1}^{p} m_i = m \tag{5}$$

The well known optimal control law is given by

$$u(t) = -R^{-1}B'P(t)X_o \tag{6}$$

where P(t) satisfies the Riccati matrix differential equation

$$\dot{P}(t) = -A'P(t) - P(t)\mathbf{A} + P(t)BR^{-1}B'P(t) - Q \tag{7}$$

A solution P(t) to (7) taking the form (3) will be shown in the following sections to lead to the local optimal feedback control law

$$(Bu)_i = (\; BR^{-1}B')_i \; P_i \quad x_i \qquad\qquad i=1,2,\ldots,k \tag{8}$$

Where $(\cdot)_i$ denotes the component of the matrix appearing in the (i,i) position. Notice that no block-diagonal form has been assumed for B and R. In the case that full state feedback may not be assumed, one can take Q in (4b) to be

$$Q = C'\tilde{Q}C \tag{9}$$

where $y = Cx$ is the output and $\tilde{Q}$ is the weight given to the output y. For $ZZ'=C'\tilde{Q}C$ it is assumed that (A,Z) is observable and (A,B) controllable. This problem shall be used as motivation for what follows.

As can be seen from (7), one may consider the flow of the Riccati differential equation to be completely specified by the 5-tuple (A,B,Q,R,S) where S has been taken here to be the zero matrix. This paper will, therefore, concentrate its study on this 5-tuple.

The problem will now consist of finding conditions on the 5-tuple (A, B,Q,R,S) so that the optimal solution will take the form (8). In order to obtain (3) from a given equation (7), it will be necessary to transform the 5-tuple (A,B,Q,R,S) into an equivalent system by an element of a group of transformations (to be defined later) denoted by $\tilde{R}$. It will be assumed that the transformations used do not affect the system's characteristics which are of interest in the models application. For this purpose two 5-tuples will be said to be equivalent, or similarly, to give rise to equivalent optimal control problems if one may be obtained from the other through a transformation by an element of $\tilde{R}$. These will then define the group of "allowable" transformations.

III. TRANSFORMATIONS ON THE 5-TUPLE

The action of the feedback group F, which has been well studied in the literature [3], is defined on pairs of matrices (A,B) arising from a dynamical system of the form (1b). There are three different actions of F on the pair (A,B), these are

$$1 - (A,B) \overset{T}{\to} (TAT^{-1},TB) \qquad |T| \neq 0 \tag{10a}$$

$$2 - (A,B) \overset{V}{\to} (A,BV^{-1}) \qquad |V| \neq 0 \tag{10b}$$

$$3 - (A,B) \overset{L}{\to} (A-BL,B) \qquad L \text{ arbitrary} \tag{10c}$$

where the constant matrices T,V and L are assumed to be of appropriate dimensions. Action 1 represents a change of basis in the state space, action 2 represents a change of basis in the input space, and the last action arises from linear feedback. The optimal control problem, however, is defined by the 5-tuple (A,B,Q,R,S). Expanding these actions to the 5-tuple (A,B,Q,R,S) generates the following set of transformations:

$$1 - (A,B,Q,R,S) \overset{T}{\to} (TAT^{-1},TB,T'^{-1}QT^{-1},R,T'^{-1}S) \qquad |T| \neq 0 \tag{11a}$$

$$2 - (A,B,Q,R,S) \overset{V}{\to} (A,BV^{-1},Q,V'^{-1}RV^{-1},SV^{-1}) \qquad |V| \neq 0 \tag{11b}$$

$$3 - (A,B,Q,R,S) \overset{L}{\to} (A-BL,B,Q+L'RL-L'S-SL,R,S-L'R) \qquad L \text{ arbitrary} \tag{11c}$$

where again, the matrices T,V, and L are taken to be of appropriate dimensions. There is a fourth action which is not available for the feedback group $\mathcal{F}$ but may be added to equations (11). This action arises from a change of variables in equation (7) and is given by

$$\bar{P} = P + M \tag{12}$$

and gives rise to the fourth action

$$4 - (A,B,Q,R,S) \overset{M}{\rightarrow} (A,B,Q-A'M-MA,R,S-MB) \qquad M = M' \tag{11d}$$

Actions 1-4 of equations (11) forms what is termed in [1] the Riccati group $\tilde{\mathcal{R}}$. As previously mentioned, two 5-tuples (A,B,Q,R,S) and $(\tilde{A},\tilde{B},\tilde{Q},\tilde{R},\tilde{S})$ will be called equivalent if there exists a set of transformations of the form (11) taking one 5-tuple into the other.

Keeping in mind that the purpose of this analysis is to study the flow of the Riccati differential equation (7) it becomes obvious that it is not necessary to study 5-tuples (A,B,Q,R,S) but infact only 3-tuples of the form $(A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S')$. On a single orbit, the action of $\tilde{\mathcal{R}}$ is transitive, and so this gives rise to a partition of $\{(A,B,Q,R,S)\}$ into systems of imprimitivity for $\tilde{\mathcal{R}}$ [8], and thus to an imprimitive action. The action of $\tilde{\mathcal{R}}$ on these 3-tuples is given by

$$\begin{aligned}
&1 - (A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S') \overset{T}{\rightarrow} (TAT^{-1}-TBR^{-1}S'T^{-1}, TBR^{-1}B'T', \\
&\qquad\qquad T'^{-1}QT^{-1}-T'^{-1}SR^{-1}S'T^{-1}) \qquad |T| \neq 0 \\
&2 - (A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S') \overset{V}{\rightarrow} (A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S') \\
&\qquad\qquad |V| \neq 0 \\
&3 - (A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S') \overset{L}{\rightarrow} (A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S') \\
&\qquad\qquad L \text{ arbitrary} \\
&4 - (A-BR^{-1}S, BR^{-1}B', Q-SR^{-1}S') \overset{M}{\rightarrow} (A-BR^{-1}S'-BR^{-1}B'M, BR^{-1}B', Q-A'M-MA-SR^{-1}S' \\
&\qquad\qquad +SR^{-1}B'M+MBR^{-1}S'-MBR^{-1}B'M)
\end{aligned} \tag{13}$$

As can be seen from equations (13), actions 2 and 3 have no effect on the 3-tuple and are therefore unnecessary. The group of transformations considered in the rest of this paper will be that given by actions 1 and 4 and will be denoted by $\mathcal{R}$.

IV. ACTIONS ON THE HAMILTONIAN

As is well known, the Riccati differential equation

$$\dot{P}(t) = -A'P(t) - P(t)A + (P(t)B+S)R^{-1}(P(t)B+S)' - Q \tag{14}$$

may be related to the solution of the Hamiltonian system

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} (A-BR^{-1}S') & -BR^{-1}B' \\ -(Q-SR^{-1}S') & -(A-BR^{-1}S')' \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{15}$$

Equation (15), or actually its flow, is known to evolve on a nonlinear surface [5] called the Grassmann manifold. Thus the nonlinearity which seems to have been lost in going from (14) to (15) is contained in the surface on which (15) evolves. The solution to (14) is obtained from that of (15) by the linear fractional transformation

$$P(t) = (\phi_{21} + \phi_{22}P_o)(\phi_{11} + \phi_{12}P_o)^{-1} \tag{16}$$

where

$$P_o = P(\dagger) \tag{17}$$

is boundary condition for (14), and

$$\Phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \tag{18}$$

is the transition matrix for (15). It is assumed that the inverse in (16) exists.

Since equation (15) characterizes the flow of the Riccati differential equation (14) and is linear, thus simpler to work with, the rest of the analysis will be carried out on this matrix. Notice that just as in the last section we need only consider 3-tuples $(A-BR^{-1}S', BR^{-1}B', Q-SR^{-1}S')$ to characterize (15).

To simplify the calculations involved, it will be assumed henceforth that S is the zero matrix, so that equation (15) becomes

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B' \\ -Q & -A' \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{19}$$

and the 3-tuples treated will be of the form $(A, BR^{-1}B', Q)$.

The first step towards translating the analysis of the last section to the Hamiltonian system is to find out how the Riccati group $\mathcal{R}$ acts on (19). In order to do this, the following definition is necessary.

Definition 1 - Let X be a 2n-dimensional square matrix. X is said to be symplectic, denoted $X \varepsilon Sp(2n)$, if

$$X'JX = J \tag{20a}$$

where

$$J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} \tag{20b}$$

and I is the n-dimensional identity matrix.

Definition 2 - Let Y be a 2n-dimensional square matrix. Then Y is said to be Hamiltonian (or infinitesimally symplectic), denoted $Y \varepsilon sp(2n)$ if

$$Y'J + JY = 0 \tag{21}$$

Remark 1 - i) The symplectic matrices form a group under matrix multiplication called the symplectic group. This is in fact a Lie group. The Hamiltonian matrices sp(2n) form the Lie algebra of Sp(2n).

ii) Given any Hamiltonian matrix H, $X \varepsilon Sp(2n)$ acts on H by the rule

$$(H, X) \longmapsto X^{-1}HX \tag{22}$$

iii) Note that (19) is indeed Hamiltonian by this definition.

It is now possible to state the following theorem.

Theorem 1 - The Riccati group as defined by its action on the Hamiltonian (22) forms a subgroup of the symplectic group. Its matrix representation is given by:

I - Change of basis in the state space by a nonsingular matrix T has representation

$$\begin{bmatrix} T & O \\ O & T^{-1'} \end{bmatrix}$$

II - Change of basis in the input space by a nonsingular matrix V has representation

$$\begin{bmatrix} I & O \\ O & I \end{bmatrix}$$

III - Linear feedback by an arbitrary matrix L has representation

$$\begin{bmatrix} I & O \\ O & I \end{bmatrix}$$

IV - Change of variables in the Riccati differential equation by an arbitrary symmetric matrix M has representation

$$\begin{bmatrix} I & O \\ -M & I \end{bmatrix}$$

As noted in the last section for the case of 5-tuples, actions II and III have no effect on the Hamiltonian matrix. The following is a straight forward result of this theorem.

Corollary 1 - The flow of the matrix Riccati differential equation

$$\dot{P}(t) = -A'P(t) - P(t)A + P(t)BR^{-1}B'P(t) - Q$$

is invariant under change of basis in the input space and linear feeback.

Thus the flow of the Riccati differential equation describes the flow to an entire class of systems. In particular, all linear feedback equivalent systems give rise to the same equation, and thus have the same flow. Once again, the Riccati group $\mathcal{R}$ will be formed only by actions I and IV. The next corollary follows from theorem 1 and a theorem in [9].

Corollary 2 - The Riccati group contains all lower block-triangular symplectic matrices.

From this discussion, the problem now becomes that of finding conditions under which given the Hamiltonian matrix H in (19) there exists a matrix $\psi \varepsilon Sp(2n)$ so that

$$\psi^{-1}H\psi = \tilde{H} \tag{23}$$

with $\tilde{H}$ giving the block diagonal form (3) to the solution of the associated Riccati differential equation by way of (16). Clearly, ψ will be of the form

$$\psi = \begin{bmatrix} T & O \\ -MT'^{-1} & T'^{-1} \end{bmatrix} \tag{24}$$

where T is a nonsingular matrix and M a symmetric matrix of appropriate dimensions. As already stated, H and $\psi^{-1}H\psi$ will be considered to give rise to equivalent optimal control problems of the form (4).

In order to solve this problem it is necessary to find conditions on $(A, BR^{-1}B', Q)$ defining the Hamiltonian (19) so that the solution P(t) to (7) has block diagonal form. Clearly, if the ϕ_{ij} in (16) are block-diagonal along with P_o then P(t) will be block diagonal (where blocks of ϕ_{ij} and P_o are of appropriate dimensions so that their combination in (16) remains block diagonal).

It is known [4] that given any Hamiltonian matrix H, there exist a symplectic matrix θ so that

$$\tilde{H} = \theta^{-1}H\theta = \begin{bmatrix} H_{11} & & 0 & H_{21} & & 0 \\ & \ddots & & & \ddots & \\ 0 & & H_{1k} & 0 & & H_{2k} \\ H_{31} & & 0 & H_{41} & & 0 \\ & \ddots & & & \ddots & \\ 0 & & H_{1k} & 0 & & H_{4k} \end{bmatrix} \qquad (25)$$

where $\tilde{H}$ is in canonical form. This however requires an action by the full symplectic group.

LEMMA 1 - Let H be a given Hamiltonian of the form (19) with no zero or purely imaginary eigenvalues. Then H can be transformed into the canoncial form (25) by an action of the Riccati group only if $BR^{-1}B'=0$.

Since the actions must be restricted to those of the Riccati group, it will not in general be possible to obtain this canonical form. The obstacle to obtaining (25) as can be observed from the lemma, comes from the $BR^{-1}B'$ term. In this case, in which H has no zero or pure imaginary eigenvalues, it is shown in [4] that (25) has $H_{2j}=0$ $j=1,2,\dots,k$. This means that $PBR^{-1}B'P'$ must be zero for some nonsingular matrix P if (25) is to be possible using only the Riccati group. This then implies that $BR^{-1}B'=0$.

In order to investigate the action of $\mathcal{R}$ on the Hamiltonian H, it is reasonable to study those systems which give an orbit under $\mathcal{R}$ of maximal dimension, that is, those systems for which the stabilizer has minimal dimension. This represents the best situation for obtaining a maximal decomposition under $\mathcal{R}$. The following result defines the stabilizer of the action.

LEMMA 2 - The stabilizer of

$$H = \begin{bmatrix} A & -BR^{-1}B' \\ -Q & -A' \end{bmatrix}$$

under the action of $\psi \varepsilon \mathcal{R}$ is given by

$$\text{sTa. } O_{\mathcal{R}}(H) = \{\psi = \begin{bmatrix} P & 0 \\ -P'^{-1}M & P'^{-1} \end{bmatrix} \mid |P| \neq 0,\ M=M', \text{ and } P \text{ and } M \text{ satisfy } (26)\}$$

where equations (26) are

$$A = PAP^{-1} + BR^{-1}B'\ M \qquad (26a)$$

$$BR^{-1}B' = PBR^{-1}B'P' \qquad (26b)$$

$$MA + A'M - MBR^{-1}B'M + Q - P'^{-1}QP^{-1} = 0 \qquad (26c)$$

In order to find sufficient conditions for the stabilizer to have minimal dimension, the following technical lemma is needed.

LEMMA 3 - Let R and Q be two given nxn-dimensional matrices with $Q=Q'$, and $R=R'$ positive definite. Suppose that all eigenvalues of $Q-\lambda R$ are distinct. Then

$$P'RP=R \qquad P'QP'=Q$$

if and only if

$$P = \psi^{-1}\Omega\psi \qquad \Omega \in \left\{\begin{bmatrix} \pm 1 & & 0 \\ & \pm 1 & \\ 0 & & \ddots & \\ & & & \pm 1 \end{bmatrix}\right\}$$

$\psi \varepsilon G\ell(n)$ is given by $\psi R\psi'=I$ and $\psi Q\psi'=\text{diag.}\{q_i\}$.

Clearly this set has dimension zero. It is now possible to state theorem 2, namely:

Theorem 2 - Given the differential matrix Riccati equation

$$\dot{P}(t) = -A'P(t) - P(t)A + P(t)BR^{-1}B'P(t) - Q$$

with associated Hamiltonian differential equation

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B' \\ -Q & -A' \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}$$

a sufficient condition for $\mathcal{O}_{\mathcal{R}}(H)$ (the orbit of the Hamiltonian matrix H under action by $\mathcal{R}$) to have maximal dimension is that

$$|BR^{-1}B'| \neq 0$$

and

$$(A'(BR^{-1}B')^{-1}A + Q) + \lambda \ (BR^{-1}B')$$

has distinct eigenvalues (with λ a real parameter).

The following corollary gives conditions under which the converse is true, in its statement, $[A,B]$ stands for the Lie bracket of the matrices A and B, i.e. $[A,B] = AB-BA$.

Corollary 3 - If $[A, BR^{-1}B'] = 0$, then the sufficient conditions of theorem 2 are also necessary.

The converse of theorem 2 is not, however, true in general as the next lemma shows.

LEMMA 4 - Let (A,b) be controllable. Then for

$$H = \begin{bmatrix} A & -bR^{-1}b' \\ -Q & -A' \end{bmatrix}$$

stab. $\mathcal{O}_{\mathcal{R}}(H)$ is of zero dimension.

Here we let A and Q be as before, b an nxl vector and R a non-zero scalar. In this case $|bR^{-1}b'|=0$. Thus, generically (A,b,Q,R) has orbits of maximal dimension. As a matter of fact, the stabilizer for (A,b,Q,R) is

$$\left\{ \begin{bmatrix} P & 0 \\ -P'^{-1}M & P'^{-1} \end{bmatrix} \;\middle|\; P = \pm I \quad \text{and} \quad M = 0 \right\} \tag{27}$$

V. CONCLUDING REMARKS

It should be noted that the conditions of theorem 2 are quite restrictive, in fact, it requires $m \geq n$ and B of full rank. The following lemma gives special classes of systems under which a decomposition is possible.

LEMMA 5 - Let $BR^{-1}B'>0$, $Q=Q'$ and A be a block-diagonal matrix, all of dimension nxn. If for $P \varepsilon Gl(n)$ satisfying $PBR^{-1}B'P'=I$ and $P'^{-1}QP^{-1}=\text{diag.}\{q_i\}$ it holds that $[P,A] = 0$. Then there is a $\psi \varepsilon \mathcal{R}$ so that

$$\psi^{-1} \begin{bmatrix} A & -BR^{-1}B' \\ -Q & -A' \end{bmatrix} \psi = \begin{bmatrix} A & I \\ \text{diag.}\{q_i\} & -A' \end{bmatrix} \tag{28}$$

In particular if $A=\alpha I$, for some $\alpha \varepsilon Re$, then there always exists such a ψ.

Here by diag. $\{\cdot\}$ is meant a diagonal matrix. For the type of systems in this lemma it is possible to obtain a block-diagonal solution P(t) to the Riccati differential equation whose blocks can be as small as those of A (depending on the boundary condition). Although, lemma 5 places no restrictions on the eigenvalues, the M used for the transformation is M=0. Thus the action in (28) is actually just the natural Gl(n) action on 3-tuples (A,Z_1,Z_2) where $Z_1Z_1'=BR^{-1}B'$ and $Z_2'Z_2=Q$ given by

$$(A,Z_1,Z_2) \xrightarrow{T} (TAT^{-1},TZ_1,Z_2T^{-1}) \qquad T \varepsilon G\ell(n) \tag{29}$$

and not an action by the full Riccati group.

If each of the matrices in the 3-tuple $(A,BR^{-1}B',Q)$ is in block-diagonal form, then not only will P(t) be in block-diagonal form (for suitable boundary condition) but so will $BR^{-1}B'$ and thus the optimal control law (8).

In conclusion, an approach for studying the solution structure of the Riccati differential equation, under variations on the coefficient matrices, has been presented. One step towards continuing this approach will be to develop canonical forms for the Hamiltonian under action by $\mathcal{R}$. This should lead to a better understanding of the role played by theorem 2 and the requirements necessary on $(A,BR^{-1}B',Q)$ to obtain a block-diagonal solution P(t).

REFERENCES:

1. Casti, J. - "Invariant Theory, The Riccati Group, and Linear Control Problems", IIASA RM-76-59, July 1976.
2.   - "The Linear Quadratic Control Problem: Some Recent Results and Outstanding Problems", SIAM Review, Vol. 22, No. 4, pgs. 459-485, October 1980.
3.   - Dynamical Systems and their Applications: Linear theory, Academic Press, 1977.
4. Laub, A.J. and Meyer, K. - "Canoncial Forms for Symplectic and Hamiltonian Matrices", Celestial Mechanics 9, pgs. 213-238, 1974.
5. Martin, C.F. and Dollin B.F. - "Elements of Global Differential Geometry", preprint.
6. Robinson, R.C. - Lectures on Hamiltonian Systems, Guanabara, 1971.
7. Sandell, N.R. , Varaiya, P., Athans, M. and Sanfonor, M.G. - "Survey of Decentralized Control Methods for large Scale Systems", IEEE Trans. on Automatic Control, AC-23, No. 2, pgs. 108-128, April 1978.
8. Suprunenko, D.A. - Matrix Groups, AMS, 1976.
9. Wallach, N.R. - Symplectic Geometry and Fourier Analysis, Math Sci Press.
10. Williamson, J. - "On the Algebraic Problem Concerning the Normal Forms of Linear Dynamical Systems", American Journal of Mathematics 58, pgs. 141-163, 1936.

OPTIMAL QUANTIZED CONTROL

THOMAS R. FISCHER*
Department of Electrical Engineering
Texas A&M University
College Station, Texas 77843

ABSTRACT

The optimal closed-loop quantized control is derived for the linear-quadratic-Gaussian formulation and shown to be separable in estimation, control and quantization. In the scalar control case the optimal quantizer is the minimum mean square error quantizer; in the vector case the optimal quantizer minimizes a quadratic distortion measure with weighting matrix dependent upon the solution to the matrix Riccati equation. The optimal quantizer is time-varying and depends on the optimal (Kalman filter) state estimate and covariance, and in the scalar control case can be adapted based on a linear transformation of a normalized quantizer design. The optimal cost-to-go is shown to be the sum of the cost-to-go for the optimal continuous control solution and a term reflecting the quantizer distortion.

I. INTRODUCTION

Implicit in most approaches to the analysis and synthesis of control systems is the assumption that the resulting control can assume one of a continuum of values; this assumption is basic to the variational approach to optimization. Instrumentation of control laws, particularly using finite word length digital processors, often forces some amount of truncation or quantization of the control actually employed. Characterization of the effect of such finite word length implementations has provided a prime motivation in the past for examining quantized control. By using sufficiently large word length processors, the quantization effects due to finite word length can usually be made negligible. Corresponding instrumentation costs tend to increase rapidly with processor capability. However, with the increasing sophistication and decreasing cost of the digital processors available, word length effects alone may not be an important motivation for considering quantization.

In a number of applications, such as remote control, control of a large system, or decentralized control, quantization may occur as an intricate part of the system structure as opposed to a by-product of the instrumentation. Especially in decentralized or remote control, the controller may be physically delocalized from the plant, requiring that all control information be transferred by means of a communication channel. A digital communication system for transferring control information provides a natural motivation for considering quantized control. Moreover, information theory [1] and rate-distortion theory [2] provide fundamental limits to the transfer of continuous-valued source information to a destination.

If the effect of quantization on the system is important, it is desirable as a design and analysis methodology to incorporate within the system model considered, a quantization description that is both mathemetically

---

* This work was supported by NSF Grant No. ECS-8106699.

tractable and accurate. Unfortunately, the combination of these requirements has proven difficult to obtain. Early quantization studies [3] considered the quantizer output as the sum of the input and a "quantization noise" (see Figure 1). Often this noise has been modeled as both white and independent [4]. Other work on quantization focused on the design of optimal quantizers that minimize the distortion between quantizer input and output [5, 6]. The most prevalent quantizer studied has been the minimum mean square error (MMSE) quantizer, due to the limited amount of mathematical tractability. It is easily shown that the MMSE quantizer output and noise are orthogonal, but the noise and input are always non-positively correlated [7].

Early studies of quantized control were performed by Tou [8] and Lewis and Tou [9]. By incorporating dynamic programming with an exhaustive search for the optimal quantizer parameters, general algorithms for calculating optimal quantized controls were developed. The general approach suffers greatly from the extreme computational burden of searching for the quantizer parameters and is infeasible for most problems of three of more dimensions.

Larson [10] has considered quantized control in a linear-quadratic-Gaussian (LQG) system. The main result of this work was an extended "separation theorem", indicating that jointly optimal estimation, control and quantization may be computed by separately performing estimation and control (as in the normal separation theorem) and appending a quantizer to the controller, with parameters selected from essentially static considerations. Subsequent comment [11] and reply [12] to this work state that Larson's result requires a time-varying quantizer and is optimal in the open-loop feedback sense, but not in the closed-loop sense.

Additional work on quantization has been done by Curry [13]. In this work, algorithms were developed for optimal estimation given quantized linear state measurements. It is important to note that the emphasis of this work was on determining optimal estimation algorithms for given quantizers (a difficult problem in its own right), rather than the selection of optimal quantizers.

The main result of the present work is to derive the optimal vector quantized control solution for the LQG problem formulation. It is shown that the optimal control is separable and hence the conclusion reached in the reply by Larson and Ese [12] is incorrect. Explicit representation of the optimal cost-to-go for the quantized control solution and the difference between the cost-to-go for the quantized control and continuous control solutions are developed, providing to some extent the analytical tractability desirable for synthesis and analysis. A discussion of the implications of the time-varying quantizer is also provided.

II. MINIMUM DISTORTION QUANTIZATION

Let x be a scalar random variable with probability density function p(x). An L-level minimum mean square error quantizer with output values $y \varepsilon \{y_i\}_{i=1}^{L}$ may be designed by the variational approach of Max [7]. The resulting design equations are

$$x_i = (y_i + y_{i+1})/2, \; i = 1, 2, \ldots, L-1 \qquad (1)$$

$$y_i = \left(\int_{x_{i-1}}^{x_i} x\, p(x)dx \right) \Big/ \int_{x_{i-1}}^{x_i} p(x)dx, \quad i = 1, 2, \ldots, L \qquad (2)$$

where $\{x_i\}_{i=0}^{L}$ partition the real line and quantizer output y_i is the conditional mean of x in the interval (x_{i-1}, x_i). Equations (1) and (2) must in general be solved iteratively to obtain the optimal quantizer. Quantizer distortion criteria other than mean square error are of course possible. However, because of the importance of the MMSE quantizer and it's vector generalization in the quantized control solution in Section III, other quantizer distortion criteria will not be discussed.

In the vector case, a generalization of (1) and (2) may be easily derived. Let $\underline{x}$ be an n-dimensional, real, random vector. As a generalization of the MSE criteria, the vector quantizer is to be designed with L output values $\{\underline{y}_i\}_{i=1}^{L}$, to minimize the distortion criteria

$$J_A = E\{||\underline{x} - \underline{y}||_A^2\} \qquad (3)$$

where A is a positive definite weighting matrix and $||\underline{x}||_A^2 = \underline{x}^T A \underline{x}$. The optimum partitioning of R^n for output $\underline{y}_i$ is given by (assuming $\underline{y}_i \neq \underline{y}_j$, $j=1, \ldots n, j \neq i$)

$$q(\underline{y}_i) = \{\underline{x}\colon J_A(\underline{y}_i) \leq J_A(\underline{y}_j), \text{ for all } j=1, \ldots n; \ j \neq i\}. \qquad (4)$$

Substituting (3) into (4) and simplifying gives

$$q(\underline{y}_i) = \left\{ \underline{x}\colon \tfrac{1}{2}||\underline{y}_i - \underline{y}_j||_A - \left(\frac{\underline{y}_i - \underline{y}_j}{||\underline{y}_i - \underline{y}_j||_A}, \underline{x} - \underline{y}_j \right)_A \leq 0 \quad \text{for all } j=1, \ldots, n; \ j \neq i \right\} \qquad (5)$$

where the weighted inner product is $(\underline{x}, \underline{x})_A = ||\underline{x}||_A^2$. The boundary of region $q(\underline{y}_i)$ is given by equality in (5) and, for A=I, is seen to be the set of all $\underline{x} \in R^n$ such that the orthogonal projection of the vector $\underline{x} - \underline{y}_j$ onto the unit vector between $\underline{y}_i$ and $\underline{y}_j$ is precisely one half the distance between $\underline{y}_i$ and $\underline{y}_j$. This is clearly a generalization of (1).

Minimizing (3) with respect to $\underline{y}_i$ implies

$$\underline{y}_i = E[\underline{x} | \underline{x} \in q(\underline{y}_i)]. \qquad (6)$$

Equations (5) and (6) must in general be solved numerically in order to design the optimal quantizer, and for high dimensional vectors this can be a difficult problem. An alternative approach to quantizer design is to use an iterative design algorithm as in [14]. It should be noted that the design obtained using this algorithm is locally optimum. No general sufficiency conditions, other than the rate-distortion bounds, presently exist for an optimal vector quantizer.

For a Gaussian input vector with positive definite weighting matrix A and nonsingular covariance matrix Λ, the minimum distortion quantizer can always be related to a minimum distortion quantizer with either A=I and $\Lambda = B_d$ or $A=B_d$ and $\Lambda=I$, where B_d is a positive definite diagonal matrix. The quantizer distortion and output values are described by the following.

Lemma 1. Let $\underline{x}$ be a Gaussian random vector with mean $\hat{\underline{x}}$ and nonsingular covariance matrix Λ, and let A be a symmetric, real, positive definite matrix. Then the quantizer with output values $\underline{y} \in \{\underline{y}_i\}_{i=1}^{L}$ that minimizes the distortion $E\{||\underline{x} - \underline{y}||_A^2\}$ has output and distortion values given by either

$$\text{a)}\ \underline{y}_i = \hat{\underline{x}} + P_1^{-1}\, \underline{\beta}_{1_i},\ i = 1, \ldots, L$$

$$E\{||\underline{x} - \underline{y}||_A^2\} = \mathrm{tr}[B_d(I - \mathrm{cov}\ \underline{\beta}_1)]$$

or

$$\text{(b)}\ \underline{y}_i = \hat{\underline{x}} + P_2^{-1}\, \underline{\beta}_{2_i},\ i = 1, \ldots, L$$

$$E\{||\underline{x} - \underline{y}||_A^2\} = \mathrm{tr}[B_d - \mathrm{cov}\{\underline{\beta}_2\}]$$

where P_1 and P_2 are nonsingular matrices depending on A and Λ, B_d is a postive definite diagonal matrix depending on A and Λ, and $\underline{\beta}_1$ and $\underline{\beta}_2$ are minimum distortion quantizer outputs for zero mean Gaussian inputs $\underline{z}_1$ and $\underline{z}_2$, with covariances I and B_d, and distortion weighting matrices B_d and I, respectively. The quantizer distortion is independent of the mean of the input.

Proof. Let $\underline{\alpha} = \underline{x} - \hat{\underline{x}}$ and $\underline{\gamma} = \underline{y} - \hat{\underline{x}}$. Then $\alpha \sim N(0, \Lambda)$ and $||\underline{x} - \underline{y}||_A^2 = ||\underline{\alpha} - \underline{\gamma}||_A^2$. Hence design of the L quantizer output values, $\{\underline{\gamma}_i\}_{i=1}^{L}$, to minimize $E\{||\underline{\alpha} - \underline{\gamma}||_A^2\}$ must result in a distortion independent of $\hat{\underline{x}}$.

From the positive definite assumptions on A and Λ, there exist nonsingular matrices P_1 and P_2 depending on A and Λ such that $P_1\ \Lambda\ P_1^T = I$ and

$P_1^{-1^T} A P_1^{-1} = B_d$, and $P_2 \Lambda P_2^T = B_d$ and $P_2^{-1^T} A P_2^{-1} = I$, where B_d is a positive definite diagonal matrix depending only on A and Λ [15]. Clearly P_1 and P_2 are related by $P_1 = B_d^{-\frac{1}{2}} P_2$. Let $\underline{z}_1 = P_1 \underline{\alpha}$ and $\underline{z}_2 = P_2 \underline{\alpha}$. Then $\underline{z}_1 \sim N(0, I)$, $||\alpha||_A^2 = ||z_1||_{B_d}^2$, $\underline{z}_2 \sim N(0, B_d)$, and $||\alpha||_A^2 = ||\underline{z}_2||_I^2$. Let $\underline{\beta}_1 = P_1 \underline{\gamma}$ and $\underline{\beta}_2 = P_2 \underline{\gamma}$. Then minimizing $E\{||\underline{x} - \underline{y}||_A^2\}$ is equivalent to minimizing $J_1 = E\{||\underline{z}_1 - \underline{\beta}_1||_{B_d}^2\}$ or $J_2 = E\{||\underline{z}_2 - \underline{\beta}_2||_I^2\}$. Assume that the L-level minimum distortion quantizer is designed, with output values $\{\underline{\beta}_{1_i}\}_{i=1}^L$ for J_1 and $\{\underline{\beta}_{2_i}\}_{i=1}^L$ for J_2. Then the corresponding output values for $\underline{y}$ are given by $\underline{y}_i = \hat{\underline{x}} + P_1^{-1} \underline{\beta}_{1_i}$, i = 1, ..., L or by $\underline{y}_i = \hat{\underline{x}} + P_2^{-1} \underline{\beta}_{2_i}$, i = 1, ..., L.

The quantizer distortion is easily computed as

$$E\{||x - y||_A^2\} = tr[A(cov[\underline{x}] - cov(\underline{y})]$$

$$= tr[B_d(I - cov[\underline{\beta}_1])] \text{ for } J_1$$

$$= tr[B_d - cov[\underline{\beta}_2]] \text{ for } J_2.$$

QED

In the scalar case the weighting matrix factors from the MSE expression and the optimal L-level quantizer for arbitrary mean and variance is obtained from a single design by appropriate scaling and shifting of the output values and partitions. In the vector case for arbitrary weighting matrix A and input covariance Λ, it is clear that the optimal quantizer can be designed by a linear transformation (and scaling by the mean) of the design obtained for the more generic problem of either Λ = I and A a diagonal weighting matrix or Λ a diagonal matrix and A = I. Unfortunately, further simplification as was obtained in the scalar case is not possible. The vector quantizer must be completely redesigned for each A or Λ.

III. PROBLEM FORMULATION AND SOLUTION

A block diagram of the basic quantized control problem is shown in Figure 2. The linear plant and measurement system are each subject to random disturbances and a quantized control is to be selected to minimize a quadratic performance measure. The system equations are

$$\underline{x}_{k+1} = \Phi_k \underline{x}_k + \Gamma_k \underline{u}_k + \underline{w}_k \tag{7}$$

$$\underline{z}_k = H_k \underline{x}_k + \underline{v}_k \tag{8}$$

$$J = E\{\underline{x}_N^T Q_N \underline{x}_N + \sum_{k=0}^{N-1} (\underline{x}_k^T Q_k \underline{x}_k + \underline{u}_k^T R_k \underline{u}_k)\} \tag{9}$$

where R_k and Q_k are positive definite and positive semidefinite matrices, respectively, and $\underline{w}_k$, $\underline{v}_k$, and $\underline{x}_0$ are white Gaussian random vectors with means $\underline{0}$, $\underline{0}$, and $\underline{x}_0$, and covariances W_k, V_k, and $P(0/-1)$, respectively. Define the information available at time k to be

$$I_k = \{z_k, z_{k-1}, \ldots, z_0; u_{k-1}, u_{k-2}, \ldots\}.$$

At each time k, the class of admissible control laws are maps from I_k to the finite set of L possible (vector) controls, $U_k = \{u_{k_i}\}_{i=1}^{L}$. Selection of the optimal control involves specifying both the mapping $u_k^*(I_k)$ and the set U_k.

Interpreting (as motivated in the introduction) the quantization restriction as a "communication link" between controller and plant, implies the presence of another communication link in the (feedback) observation path. This complication is not treated in the present work and the feedback path is considered as ideal (i.e. continuous or unquantized). Note that with the measurement system as in Figure 2, because of the conditioning on information I_k, the optimal state estimate, $E[\underline{x}_k|I_k]$, is obtained by the standard Kalman filter, and all random vectors have Gaussian conditional distributions. This observation was made in [2].

The main result is given by the following.

Theorem 1. For the linear plant model in (7), observations in (8), and quadratic cost in (9), the optimal closed-loop L-level quantized control is obtained by *separately*

(a) performing the optimal state estimation (Kalman filtering) to produce $\hat{\underline{x}}_k = E[\underline{x}_k|I_k]$,

(b) generating the optimal continuous-valued control solution, $\underline{u}_k^c = -[R_k + \Gamma_k^T M_{k+1}\Gamma_k]^{-1}\Gamma_k^T M_{k+1}\Phi_k\hat{\underline{x}}_k$ to the standard LQG problem, using (a), and

(c) quantizing the continuous-valued control in (b) with the minimum distortion, L-output valued quantizer with distortion weighting matrix $A_k = [R_k + \Gamma_k^T M_{k+1}\Gamma_k]$, to produce the quantized control applied to the plant.

The set of optimal quantizer output values, $\{\underline{u}_{k_i}^*\}_{i=1}^{L}$, are chosen to minimize $E\{||\underline{u}_k^c - \underline{u}_k||_{A_k}^2\}$. The optimal cost-to-go at time k is given by

$$J_k^* = J_k^c + E\{\sum_{i=k}^{N-1} ||\underline{u}_i^c - \underline{u}_i||_{[R_i + \Gamma_i^T M_{i+1}\Gamma_i]} | I_k\}$$

where J_k^c is the optimal cost-to-go at time k for the optimal continuous-valued control solution, and M_k is the solution to the matrix Riccati equation.

Remark 1. The optimal quantized control solution for the LQG system takes the separable form of Figure 2. The weighting matrix of the quantizer distortion is dependent on M_{k+1}, the solution to the matrix Riccati equation.

Remark 2. In Larson's original result [10] only scalar control was considered. In this case the quantizer design criteria is the standard mean square error, since weighting the MSE criteria by a positive constant does not effect the minimization. The weighting is retained in calculating the optimal cost-to-go.

Proof of theorem 1. A dynamic programming argument will be used to compute the optimal quantized control and show that the optimal cost-to-go is quadratic in the state. At time N-1, choose the quantized control, u_{N-1}, to minimize

$$J_{N-1} = E\{\underline{x}_N^T Q_N \underline{x}_N + \underline{x}_{N-1}^T Q_{N-1} \underline{x}_{N-1} + \underline{u}_{N-1}^T R_{N-1} \underline{u}_{N-1} | I_{N-1}\}.$$

Substituting for $\underline{x}_N$ from (7) gives

$$J_{N-1} = E\{\underline{x}_{N-1}^T [\Phi_{N-1}^T Q_N \Phi_{N-1} + Q_{N-1}] \underline{x}_{N-1} + 2\underline{u}_{N-1}^T \Gamma_{N-1}^T Q_N \Phi_{N-1} \underline{x}_{N-1}$$

$$+ \underline{u}_{N-1}^T [R_{N-1} + \Gamma_{N-1}^T Q_N \Gamma_{N-1}] \underline{u}_{N-1} | I_{N-1}\} + tr[Q_N W_{N-1}].$$

Minimizing J_{N-1} reduces to minimizing the middle two terms, since the first and last terms do not depend explicitly on $\underline{u}_{N-1}$.

Define $\underline{\alpha}_{N-1} = -[R_{N-1} + \Gamma_{N-1}^T Q_N \Gamma_{N-1}]^{-1} \Gamma_{N-1}^T Q_N \Phi_{N-1} \underline{x}_{N-1}$. Then J_{N-1} can be rewritten as

$$J_{N-1} = E\{\underline{x}_{N-1}^T [\Phi_{N-1}^T Q_N \Phi_{N-1} + Q_{N-1}] \underline{x}_{N-1} - 2\underline{u}_{N-1}^T [R_{N-1}$$

$$+ \Gamma_{N-1}^T Q_N \Gamma_{N-1}] \underline{\alpha}_{N-1} + \underline{u}_{N-1}^T [R_{N-1} + \Gamma_{N-1}^T Q_N \Gamma_{N-1}] \underline{u}_{N-1} | I_{N-1}\}$$

$$+ tr[Q_N W_{N-1}].$$

Minimizing J_{N-1} with respect to $\underline{u}_{N-1}$ is equivalent to minimizing only the middle two terms. The probability density function of $\underline{\alpha}_{N-1}$ given I_{N-1} is Gaussian, with mean and covariance

$$\hat{\underline{\alpha}}_{N-1} = - [R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]^{-1} \Gamma^T_{N-1}Q_N\Phi_{N-1}E[x_{N-1}|I_{N-1}]$$

$$\mathrm{cov}[\underline{\alpha}_{N-1}] = [R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]^{-1} \Gamma^T_{N-1}Q_N\Phi_{N-1} \mathrm{cov}[x_{N-1}|I_{N-1}]$$

$$\Phi^T_{N-1}Q_N\Gamma_N[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]^{-1}.$$

Adding and subtracting the term $E\{\hat{\underline{\alpha}}^T_{N-1}[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]\hat{\underline{\alpha}}_{N-1}\}$ to J_{N-1} and simplifying gives

$$J_{N-1} = \underset{\underline{x}_{N-1}}{E}[\underline{x}^T_{N-1}[\Phi^T_{N-1}Q_N\Phi_{N-1} + Q_{N-1}]\underline{x}_{N-1} - \hat{\underline{\alpha}}^T_{N-1}[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]$$

$$\hat{\underline{\alpha}}_{N-1}|I_{N-1}\} + \underset{\hat{\underline{\alpha}}_{N-1}}{E}\{||\hat{\underline{\alpha}}_{N-1} - \underline{u}_{N-1}||^2_{[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]}\} + \mathrm{tr}[Q_NW_{N-1}].$$

Minimizing J_{N-1} with respect to u_{N-1} is equivalent to minimizing just the middle term. This is recognized as the quantizer distortion criteria for quantizer input $E[\alpha_{N-1}|I_{N-1}] = - [R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]^{-1}\Gamma^T_{N-1}Q_N\Phi_{N-1}E[\underline{x}_{N-1}|I_{N-1}]$, with output $\underline{u}_{N-1}$, and weighting matrix $[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]$. The term $E[\underline{\alpha}_{N-1}|I_{N-1}]$ is just the continuous-valued control solution for time stage N-1. Now quantizer input $\hat{\alpha}_{N-1}$ is a number for any realization of I_{N-1}. However from the viewpoint of a quantization representation of $\hat{\alpha}_{N-1}$, the quantizer must be designed based on the ensemble of z_{N-1}. Otherwise a one level quantizer could represent the number $\hat{\alpha}_{N-1}$ perfectly.

The optimal cost-to-go is calculated to be

$$J^*_{N-1} = \underset{x_{N-1}}{E}\{\underline{x}^T_{N-1}M_{N-1}\underline{x}_{N-1}|I_{N-1}\} + \mathrm{tr}[S_{N-1}\ \mathrm{cov}[\underline{x}_{N-1}]]$$

$$+ \underset{\hat{\underline{\alpha}}_{N-1}}{E}\{||\hat{\underline{\alpha}}_{N-1} - \underline{u}^*_{N-1}||^2_{[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]}\} + \mathrm{tr}[Q_NW_{N-1}]$$

where $M_{N-1} = \Phi^T_{N-1}[Q_N - Q_N\Gamma_{N-1}[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]^{-1} \Gamma^T_{N-1}Q_N] \Phi_{N-1} + Q_{N-1}$,

$S_{N-1} = \Phi^T_{N-1}Q_N\Gamma_{N-1}[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]^{-1} \Gamma^T_{N-1}Q_N\Phi_{N-1}$, and u^*_{N-1} is the minimum distortion quantizer output. Note that

$J^*_{N-1} = J^c_{N-1} + E\{||\hat{\underline{\alpha}}_{N-1} - \underline{u}^*_{N-1}||^2_{[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]}\}$, where J^c_{N-1} is the optimal cost-to-go for the continuous LQG solution.

At time N-2, quantized control $\underline{u}_{N-2}$ is to be selected to minimize

$$J_{N-2} = E\{\underline{x}^T_{N-2}Q_{N-2}\underline{x}_{N-2} + \underline{u}^T_{N-2}R_{N-2}\underline{u}_{N-2} + J^*_{N-1}|I_{N-2}\}.$$

It is a standard result [16] that $\mathrm{cov}[x_{N-1}|I_{N-2}, \underline{u}_{N-2}]$ is independent of $\underline{u}_{N-2}$. In lemma 1 it was shown that the distortion of the minimum distortion Gaussian vector quantizer depends only on the input covariance and weighting matrix and is independent of the mean of the input. Hence the term

$$\underset{\underline{z}_{N-1}}{E}\ \{\underset{\hat{\underline{\alpha}}_{N-1}}{E}\ \{||\hat{\underline{\alpha}}_{N-1} - \underline{u}^*_{N-1}||^2_{[R_{N-1} + \Gamma^T_{N-1}Q_N\Gamma_{N-1}]}|I_{N-1}\}|I_{N-2}, u_{N-2}\} \text{ is}$$

independent of $\underline{u}_{N-2}$ also, since $\mathrm{cov}[\hat{\underline{\alpha}}_{N-1}]$ depends only on $\mathrm{cov}[x_{N-1}]$. Thus only the term $E\{\underline{x}^T_{N-1}M_{N-1}\underline{x}_{N-1}|I_{N-1}\}$ in the expression for J^*_{N-1} need be considered for the minimization of J_{N-2}. Note that this is precisely the term that appears in the standard, continuous-valued control solution.

Finally, at each time k the quantized control $\underline{u}_k$ is the output of the minimum distortion vector quantizer with weighting matrix $[R_k+\Gamma^T_kM_{k+1}\Gamma_k]$ and input the continuous-valued control solution, $\underline{u}^c_k = E[\underline{\alpha}_k|I_k]$, where M_{k+1} is the solution to the matrix Riccati equation

$$M_k = \Phi^T_k[M_{k+1} - M_{k+1}\Gamma_k[R_k+\Gamma^T_kM_{k+1}\Gamma_k]^{-1}\ \Gamma^T_kM_{k+1}]\ \Phi_k + Q_k$$

with $M_N = Q_N$.

The additional cost of quantization at any time stage is given by

$$E\{||\hat{\underline{\alpha}}_k - \underline{u}_k||^2_{[R_k+\Gamma^T_kM_{k+1}\Gamma_k]}\}.$$

QED.

IV. DISCUSSION

In [12] it was concluded for the scalar case that optimal control and quantization are not separable. This conclusion was based on the correct calculation of the optimal cost-to-go, but the representation of the effect of quantization as an untractable sum of integrals. In the present formulation this difficulty is mitigated by the use of lemma 1 in theorem 1.

From theorem 1, it is clear that the optimal quantized control requires a time-varying quantizer. In the scalar control case this quantizer can be adapted based on the readily available state estimation information and a single, L-level, MMSE quantizer design. For a vector control, equations (3) and (4) must be resolved at each time stage to obtain the optimal quantizer.

REFERENCES

[1] C. Shannon, Bell System Tech. J., v. 27, pp. 379-423, 1948.
[2] T. Berger, Rate Distortion Theory, Prentice-Hall, 1971.
[3] B. Widrow, IRE Trans. Circuit Theory, CT-3, pp. 266-276, 1956.
[4] A. Oppenheim and R. Schafer, Digital Signal Processing, Prentice-Hall Chap. 9, 1975.
[5] S. P. Lloyd, "Least squares quantization in PCM" Bell Telephone Laboratories Paper, Murray Hill, NJ, 1957.
[6] J. Max, IEEE Trans. Inform. Theory, IT-6, pp. 7-12, 1960.
[7] J. Bucklew and N. Gallagher, IEEE Trans. Inform. Theory, IT-25, pp. 365-366, 1979.
[8] J. Tou, Optimum Design of Digital Control Systems, Academic, 1963.
[9] J. Lewis and J. Tou, Trans. AIEE, v. 82, pp. 195-201, 1965.
[10] R. Larson, IEEE Trans. Auto. Contr., AC-12, pp. 162-168, 1967.
[11] R. Marleau and J. Negro, IEEE Trans. Auto. Contr., AC-17, pp. 273-274, 1972.
[12] R. Larson and E. Tse, IEEE Trans. Auto. Contr., AC-17, pp. 274-276.
[13] R. E. Curry, Estimation and Control with Quantized Measurements, Research Monograph, no. 60, The M. I. T. Press, 1970.
[14] Linde, Buzo, Gray, IEEE Trans. Comm., COM-28, pp. 84-95, 1980.
[15] B. Noble, Applied Linear Algebra, Prentice-Hall, p. 396, 1969.
[16] D. Bertsekas, Dynamic Programming and Stochastic Control, Academic Press, p. 131, 1976.

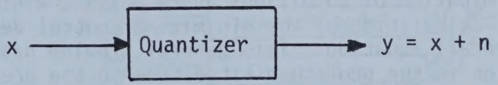


Figure 1. Quantizer additive noise model.

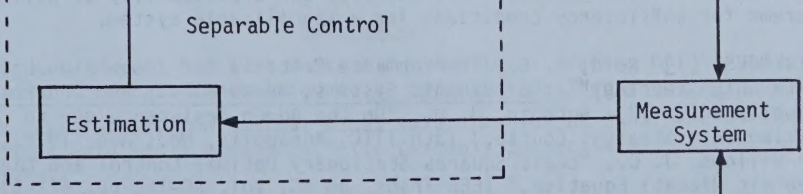


Figure 2. System configuration with general quantized control and separable implementation.

A STABLE CONTROLLER ARISING FROM MINIMIZATION OF A QUADRATIC PERFORMANCE CRITERION WITH AN INDEFINITE STATE WEIGHTING MATRIX

A. K. TUGCU and R. E. REID
University of Illinois at Urbana-Champaign
Urbana, IL 61801

ABSTRACT

The greatly increased price of fuel oil and a substantial over capacity in available tonnage of the world's merchant shipping have combined to cause a major problem in the maritime industry. Much effort is devoted to improving efficiency of ship, bridge, and propulsion systems. One important area of fuel losses occurring during the operation of a ship in the open seas has been identified as the efficiency of the autopilot for ship steering [1]. Most modern ocean-going ships are fitted with automatic steering control systems. Some attempts have been made to define a measure or index of relative steering efficiency of autopilots based on propulsion losses, e.g. [2]. Several of these were contradictory and no performance index representative of propulsion losses related to steering had been proposed which can be said to have been fully justified either theoretically or experimentally.

Recently, a performance criterion for appraisal of the efficiency of existing ship steering control systems in terms of steering-related propulsion losses was proposed by Reid [1]. However, the use of this performance criterion for design of new ship autopilots based on a linear quadratic formulation was apparently hindered by the indefinite form of the state weighting matrix. The conditions under which the existence of a Riccati gain matrix which will result in a stable closed loop system are guaranteed are well known. They include, of course, the requirement for at least positive semi-definiteness of the state weighting matrix. This condition is required to guarantee the existence of a lower bound to the performance criterion. If the state weighting is not, at least, positive semi-definite, it is not generally true that a minimum for the criterion exists. The determination of conditions under which a minimum exists as well as the actual calculation of the minimizing control depend on the solution of the Riccati equation. This general problem has commanded continuing attention in the published literature to the present time [3].

The sufficiency conditions for the guaranteed stability of the closed loop system where the state weighting matrix is indefinite based on the work of Willems [3,4] are presented. The theorem, which mainly depends on the Frequency Domain Inequality [3] is applied to the ship steering problem in the case of an open-loop-stable containership and the required existence and stability conditions are shown to be satisfied. The stability of the resulting closed loop system is verified by solution of the Riccati equation and *a posteriori* time domain simulation studies of the resulting controller/ship system. The paper therefore proves the existence of a stabilizing solution to the Riccati equation deriving from the minimum fuel optimal control application to ship steering research. The results also afford further verification of the validity and applicability of Willems' theorems for sufficiency conditions for a stabilizable system.

REFERENCES [1] Reid, R. E., "Performance Criteria for Propulsion Losses due to Ship Steering," *J. of Dynamic Systems, Measurement, and Control*, to be published., [2] Norrbin, N. H., "On the Added Resistance due to Steering on a Straight Course," 13th ITTC, Annapolis, Md., Aug. 1972., [3] Willems, J. C., "Least Squares Stationary Optimal Control and the Algebraic Riccati Equation," *IEEE Trans. on AC*, Vol. AC-16, pp. 621-634, Dec. 1971., [4] Willems, J. C., "On the Existence of a Non-Positive Solution to the Riccati Equation," *IEEE Trans. on AC*, Vol. AC-20, pp. 592-593, Oct. 1974.

A HIERARCHICAL OPTIMAL FINAL-VALUE CONTROL STRATEGY FOR NONLINEAR SYSTEMS REALIZABLE WITH SMALL COMPUTERS

B. ASSELMEYER
Coordinated Science Laboratory, University of Illinois,
1101 W. Springfield Ave., Urbana, Illinois 61801, USA
On leave from Fachgebiet Regelsystemtheorie, Technische
Hochschule Darmstadt, Darmstadt, West Germany

Final-value control means bringing a system described by a set of nonlinear ordinary differential equations

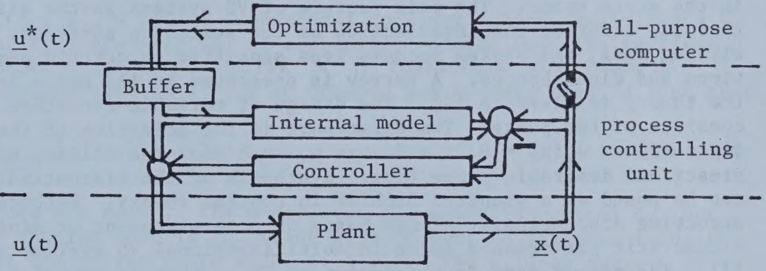
$$\dot{\underline{x}} = \underline{f}(\underline{x},\underline{u},t) \tag{1}$$

from a given initial point to a prescribed goal given by terminal conditions

$$\underline{q}(\underline{x}(t_f),t_f) = 0 \tag{2}$$

despite disturbances; it is called optimal, if $\underline{u}$ is chosen to minimize a performance criterion. This can generally be done only numerically and yields open loop control functions $\underline{u}(t)$.

For the realization of a closed loop controller a two-level strategy is applied. On the upper level an on-line optimization calculates improved control functions for the next major sampling interval. A simplified optimization procedure based on a Ritz approximation [1] of the control functions is applied. On the lower level, an explicit model-following control designed using quadratic cost criteria, counteracts disturbances already during this major sampling interval. These two levels correspond to a computer hierarchy in the controller where an all-purpose computer performs the dynamic optimization and a dedicated, microprocessor based process controlling unit serves for the actual on-line control.



Hardware simulations [2] using a model of a chemical reactor as an example show that sampling times of a few minutes for the major intervals, and of about 1 sec. for the process controlling unit can be obtained, while only microprocessor based computers are used.

References

[1] B. Asselmeyer, "Optimal Control for Nonlinear Systems Calculated with Small Computers," submitted to J. Optimization Theory and Applications.

[2] B. Asselmeyer, "On Optimal Final Value-Control of Nonlinear Systems Using only Small Computers," (in German), Darmstaedter Dissertation, 1980.

A SERVO COMPENSATOR DESIGN APPROACH FOR VARIABLE STRUCTURE SYSTEMS

A. J. CALISE AND K. V. RAMAN
Mechanical Engineering and Mechanics Department
Drexel University
Philadelphia, PA 19104

ABSTRACT

This paper presents an approach for designing a variable structure servo-mechanism for the case of constant disturbance inputs. The design makes use of servo compensators to achieve asymptotic regulation. The effect that disturbances within and outside the range space of the control action have on the design structure are separately discussed. The main result relies on a hyperstability condition which guarantees reaching and existance of the sliding mode.

1. INTRODUCTION

It is often necessary that, in solving practical tasks of automatic control, the system behavior be independent of the influence of system parameter variations and provide asymptotic regulation in the presence of external disturbances. One way to achieve this is the application of sliding modes in a system with variable structures.

A variable structure (VS) system is a feedback system with a discontinuous feedback law whose coefficients switch on hypersurfaces defined in the state space. The main feature of VS systems is the sliding motion that can occur on the intersection of the switching surfaces. While in the sliding mode, the system becomes less sensitive to certain parameter variations and disturbances. A survey is presented in [1] and a development of the theory is given in [2]. The design of variable structure systems consists of two parts. The first part is the selection of the discontinuity surfaces $s_i(x) = 0$, $i = 1 \cdots m$, such that the sliding motion has prescribed desirable properties. Synthesis of the discontinuity surfaces can be posed as a standard problem in control theory. Methods for constructing discontinuity planes based on pole placement or minimization of a quadratic performance index in multidimensional VS systems are given in [3]. The second part in the design of VS systems is choosing a feedback control law of the form

$$u_i = \begin{cases} u_i^+(x) , & s_i(x) > 0 \\ u_i^-(x) , & s_i(x) < 0 \end{cases} \tag{1}$$

to insure that a sliding motion occurs on the intersection of the discontinuity surfaces. A major aspect in the design of VS systems involves constraining the parameters of the sliding plane and the control to guarantee existence of the sliding mode, and that the sliding surface is reached from all potential initial states.

A number of techniques have been proposed to date for the design of multi-variable regulators enjoying the properties of disturbance rejection and, in some cases, structural stability. These specific design methodologies can be grouped into two distinct variations - those employing estimates of (possibly artificial) disturbance states, and those employing dynamic error augmentation. Reference [4] specifies two basic synthesis procedures which typify the essential variants to be found in the references.

Procedures for designing VS systems with the aim of disturbance rejection, while in sliding mode, were first discussed in [5] and [6]. In these earlier works, the dynamics of the disturbances are essentially unknown. Only the bounds on the disturbances and their derivatives are used in the design. However, these procedures required the use of the derivatives of the state in the design.

More recently in [7], the disturbance is modeled by the homogeneous response of a known system of linear time invariant differential equations. The resulting design procedure is analogous to the solution of the linear multivariable servo mechanism problem, which incorporates the estimates of disturbance states.

This paper deals with the development of a VS Servo Compensator based on a synthesis procedure which employs dynamic error augmentation. The VS Servo is developed for the case of constant disturbances.

2. INVARIANCE IN VS SYSTEMS

Consider the following controllable linear system

$$\dot{x} = Ax + Bu + Ew \tag{2}$$

where $x \in R^n$, $u \in R^m$, w is a q dimensional constant unknown disturbance vector, and A, B, E are matrices of appropriate dimension. The matrices B and E are assumed constant . Matrix A is considered to be made up of a known constant part (A_0) plus an unknown but bounded part δA such that

$$a_{ij_{min}} \le \delta A_{ij}(t) \ge a_{ij_{max}} \tag{3}$$

where the bounds are known. The motivation for using VS control is that the motion in sliding mode is invariant with respect to variations due to $\delta A(t)x(t)$ that lie in the range space of B [8]. The sliding surface is defined by the intersection of m hyperplanes

$$s = C\,x = 0 \quad , \quad s \in R^m \tag{4}$$

where it is assumed that rank (CB) = m. If the disturbances also lie in the range space of B, then the motion in sliding mode is characterized by an (n-m) order system

$$\dot{x}^1 = A^1 x^1 \tag{5}$$

where A^1 depends on A_0, δA^1, and C. The term δA^1 reflects the part of $\delta A(t)x(t)$ that lies outside the range space of B. To illustrate this property of a VS system, we express (without loss of generality) the dynamics in (2) in the form

$$\begin{aligned} \dot{x}_1 &= A_{11}x_1 + A_{12}x_2 + E_1 w \\ \dot{x}_2 &= A_{21}x_1 + A_{22}x_2 + B_2 u + E_2 w \end{aligned} \tag{6}$$

where $x_1 \varepsilon R^{n-m}$ and $x_2 \varepsilon R^m$. Also without loss of generality we may express (4) as

$$s = C_1x_1 + x_2 = 0 \tag{7}$$

Using (7) and the first of (6), the motion in sliding mode reduces to

$$\dot{x}_1 = (A_{11} - A_{12}C_1)x_1 + E_1w \quad , \quad x_2 = -C_1x_1 \tag{8}$$

If A_{11} and A_{12} are known, and if the x_1 equation in (6) is (A_{11}, A_{12}) controllable, then C_1 in (8) can be chosen to achieve arbitrary pole placement or to minimize a quadratic cost function. The motion in sliding mode is invariant with respect to variations in A_{21}, A_{22} and w for the case $E_1 = 0$. It is also invariant with respect to B_2.

For the case $E_1 \neq 0$, it is impossible to regulate the total state to the origin when $n > m$. In this case, we consider the problem of regulating an output

$$y = G_1x_1 + G_2x_2 \tag{9}$$

where $y \varepsilon R^p$ and $p \leq m$. This can be accomplished by introducing a servo compensator to the dynamics in sliding mode:

$$\dot{x}_1 = A_{11}x_1 + A_{12}x_2 + E_1w \tag{10,a}$$

$$\dot{\eta} = G_1x_1 + G_2x_2 \tag{10,b}$$

and regarding x_2 as a control variable of the form

$$x_2 = -C_1x_1 - C_2\eta \tag{11}$$

This implies a sliding surface of the form

$$s = C_1x_1 + C_2\eta + x_2 = 0 \tag{12}$$

Stability of the system in (10) implies

$$\lim_{t\to\infty} y(t) = 0 \tag{13}$$

In general, arbitrary pole placement, by a suitable choice of C_1 and C_2 is possible for the system in (10) if it is $[A_1, B_1]$ controllable, where

$$A_1 = \begin{bmatrix} A_{11} & 0 \\ G_1 & 0 \end{bmatrix} \quad , \quad B_1 = \begin{bmatrix} A_{12} \\ G_2 \end{bmatrix} \tag{14}$$

By a slight extension of the results in [9], the necessary and sufficient condition for $[A_1, B_1]$ controllability are:

(i) The first of (10) is $[A_{11}, A_{12}]$ controllable

$$\text{(ii) Rank}\begin{bmatrix} A_{11} & A_{12} \\ G_1 & G_2 \end{bmatrix} = n - m + p$$

3. VS CONTROL DESIGN

From the previous section it is apparent that the central issue reduces to the design of a variable structure control of the form (1) that guarantees both reaching and existence of the sliding mode. One approach is to design u such that it guarantees

$$\dot{s}^T s < 0 \quad , \quad |s| > 0 \tag{15}$$

The difficulty that arises in the presence of an external disturbance is that $\dot{s}$ is dependent on the disturbance. Thus it is necessary to either establish bounds on the disturbance, as in [8], or to estimate the disturbance using an observer, as in [7]. If disturbance bounds are used, then we lose the property that the switching level of the control reduces to zero as the desired equilibrium point is reached.

To illustrate this point, consider the system in (2) for the case of scalar control. Also, assume that the effect of the disturbance lies in the range space of B, or that

$$\text{rank}[B, E] = \text{rank}[B] \tag{16}$$

Then u can be any function that satisfies:

$$CBu^{+}(x) < -C(Ax + Ew) \quad , \quad s > 0 \tag{17}$$

$$CBu^{-}(x) > -C(Ax + Ew) \quad , \quad s < 0 \tag{18}$$

In the absence of w, we may select

$$u(x) = -\sum_{i=1}^{n} \psi_i x_i \tag{19}$$

$$\psi_i = \begin{cases} \alpha_i & , \quad x_i s > 0 \\ \beta_i & , \quad x_i s < 0 \end{cases} \tag{20}$$

with

$$\begin{aligned} CB\alpha_i &> \sup_t \left\{ CA_i \right\} \\ CB\beta_i &< \inf_t \left\{ CA_i \right\} \end{aligned} \tag{21}$$

where the A_i are the columns of A. This results in the property that $u(x) \to 0$ as $x \to 0$. In the presence of w, we must include another term in (19)

$$u(x) = - \sum_{i=1}^{n} \psi_i x_i - \psi_{n+1} \tag{22}$$

$$\psi_{n+1} = \begin{cases} \alpha_{n+1} , & s > 0 \\ \beta_{n+1} , & s < 0 \end{cases} \tag{23}$$

with

$$\begin{aligned} CB\alpha_{n+1} &> \sup_t \left\{ CEw \right\} \\ CB\beta_{n+1} &< \inf_t \left\{ CEw \right\} \end{aligned} \tag{24}$$

In this case, if C is chosen such that the sliding motion is stable, then the origin is an asymptotically stable equilibrium point, but u(x) will contain a finite switching component at the origin.

To circumvent the above problem, a second servo compensator is introduced. In this section it will be shown that the VS control can be designed as the sum of two components:

$$u = u_1 + u_2 \tag{25}$$

where u_1 has the form (1) and can be designed by disregarding the effects of disturbances, and u_2 is a linear control

$$u_2 = K_1 s + K_2 \xi \quad , \quad \xi \in R^m \tag{26}$$

where ξ is the state vector of a second servo compensator having the form

$$\dot{\xi} = s \tag{27}$$

Stability of the combined system is demonstrated using hyper-stability theory [10], where s is regarded as the output of a hyper-stable block. The resulting VS servomechanism structures for the two cases discussed in the previous section are illustrated in Figures 1 and 2. In general, the eigenvalues associated with the reaching phase are largely determined by the choice of K_1 and K_2, while the motion in sliding (as discussed previously) is determined by the choice of C_1 and C_2.

3.1 Hyperstable VS Control Design

Given a plant described in the form of (6), then the dynamics of the combined system in Figure 2 are defined by the following equations:

$$\dot{z} = \bar{A}z + \bar{B}u + \bar{E}w \tag{28}$$

$$\dot{\xi} = s = \bar{C}z \tag{29}$$

where

$$z^T = [x_1^T, x_2^T, \eta^T] \tag{30}$$

$$\bar{C} = [C_1, I, C_2] = [\bar{C}_1, C_2] \tag{31}$$

$$\bar{A} = \left[\begin{array}{c|c} A & 0 \\ \hline G & 0 \end{array}\right] , \quad \bar{B} = \left[\begin{array}{c} 0 \\ \hline B_2 \\ \hline 0 \end{array}\right] , \quad \bar{E} = \left[\begin{array}{c} E_1 \\ \hline E_2 \\ \hline 0 \end{array}\right] \tag{32}$$

and u is defined by (25) and (26). The motion normal to the subspace defined by s = 0 can be examined by considering

$$\begin{bmatrix} \dot{s} \\ \dot{\xi} \end{bmatrix} = \left[\begin{array}{c|c} B_2K_1 & B_2K_2 \\ \hline I & 0 \end{array}\right] \begin{bmatrix} s \\ \xi \end{bmatrix} + \left[\begin{array}{c} \bar{C}E \\ \hline 0 \end{array}\right] w + \left[\begin{array}{c} \dot{s}_o \\ \hline 0 \end{array}\right] \tag{33}$$

where

$$\dot{s}_o = \bar{C}\bar{A}z + B_2u_2 \tag{34}$$

To guarantee reaching and existence of a sliding mode along s = 0, it is necessary that the dynamics in (33) be at least asymptotically stable at

$$s = 0 \quad , \quad B_2K_2\bar{\xi} = \bar{C}Ew \tag{35}$$

for all s and ξ in R^{2m}, and for any z(t). In considering the equilibrium condition in (35), note that motion of (33) along s = 0 is the same as that obtained when the VS control u_2 is replaced by its equivalent control [2]

$$u_2^{eq} = -B_2^{-1}\bar{C}\bar{A}z \tag{36}$$

The motion with reference to this equilibrium condition can be defined as

$$\dot{\chi} = A_o\chi + B_o\dot{s}_o \tag{37}$$

where

$$\chi^T = [s^T, \delta\xi^T] \quad , \quad \delta\xi = \xi - \bar{\xi} \tag{38}$$

$$A_o = \left[\begin{array}{c|c} B_2K_1 & B_2K_2 \\ \hline I & 0 \end{array}\right] , \quad B_o = \left[\begin{array}{c} I \\ \hline 0 \end{array}\right] \tag{39}$$

We will regard s as an output of (37) by writting

$$s = C_o\chi \quad , \quad C_o = [1, 0] \tag{40}$$

and u_2 (z(t), s) as any nonlinear feedback satisfying

$$\eta(t_o, t_1) = \int_{t_o}^{t_1} \dot{s}_o^T s\,dt \leq \gamma_o^2 \quad \text{for all } t_1 \geq t_o \tag{41}$$

In particular, we may choose u_2 to guarantee

$$\dot{s}_o^T s < 0 \tag{42}$$

This implies that the VS design of u_2, for the system in (28), can be carried out while disregarding the disturbance term.

The following theorem summarizes the main result.

Theorem

The equilibrium state (35) for the system (33) is globally asymptotically stable for all $u_2(z(t),s)$ satisfying (41) if K_1 and K_2 are chosen such that A_o in (39) is a negative semi-definite matrix.

Proof

A necessary and sufficient condition for the system in (33) to be hyperstable is that the transfer matrix

$$H(s) = C_o(pI - A_o)B_o \tag{43}$$

be positive real (PR). Moreover, H(s) if PR if and only if there exist symetric matrices $P>0$ and $Q\geq 0$ such that

$$PA_o + A_o^T P = -Q \tag{44}$$

$$B_o^T P = C_o \tag{45}$$

The reader is referred to [11] for an excellent summary discussion of the above statements. Substituting the expressions for B_o and C_o in (45) we find that P may be chosen as the identity matrix. Hence, (44) leads to the conclusion that K_1 and K_2 should be chosen such that A_o is a negative semi-definite matrix, or that the eigenvalues of $A_o + A_o^T$ be non-positive. In general, there will be at least m eigenvalues at the origin.

3.2 Summary of the Design Procedure

The design procedure can be summarized by the following steps:

1. Select C_1 and C_2 for the system in (10) in accordance with the motion desired in sliding mode.

2. Select K_1 and K_2 such that the eigenvalues of $A_o + A_o^T$ are non-positive.

3. Design $u_2(z,s)$ as a VS control for (28) to insure that (41) or (42) is satisfied for all z(t).

In the case where $E_1 = 0$, step 1 reduces to designing C_1 for the system in (10, a). In this case, the total state is regulated to the origin, and $z(t) = x(t)$ in step 3. This implies that the design in step 3 can be carried out such that $u_2 \to 0$ asymptotically. The selection of K_1 and K_2 should be such that the transient associated with reaching is short in comparison to the response of the system in sliding mode. The motion in sliding mode is invarient with respect to unknown but bounded variations in A_{21} and A_{22}. It is also invariant with respect to variations in B_2; however, stability relies on B_2 being constant. Also, the uncertainties in B_2 must be such that the condition $A_o \leq 0$, and (41) are preserved.

4. CONCLUSION

A servo compensator design approach for VS systems has been outlined for the case of constant disturbances. The approach relies on hyperstability theory to insure global asymptotic stability of the closed loop system. The designer may separately specify the response characteristics of the reaching and sliding motions.

ACKNOWLEDGEMENT

This research is supported by the Office of Naval Research under contract number N00014-77-C-D642.

REFERENCES

1. Utkin, V.I. "Variable Structure Systems with Sliding Mode: A Survey," IEEE Trans. A.C., Vol. AC-22, pp. 212-222, April, 1977.

2. Utkin, V.I., Sliding Modes and Their Application in Variable Structure Systems, MIR Publishers, Moscow, 1974.

3. Utkin, V.I., Young, K.D., "Methods for Constructing Discontinuity Planes in Multidimensional Variable Structure Systems," Trans. from Avtomatika i Telemekhanika, No. 10 pp. 72-77.

4. Kwatny, H.G. Kalnitsky, K.C., "On Alternative Methodologies for the Design of Robust Linear Multivariable Regulators," IEEE Trans. A.C., Vol. AC-23, No. 5, October 1978, pp. 930-933.

5. Utkin, V.I., "Compensation for the Forced Component of Motion in VS Systems," Engineering Cybernetics, No. 4, pp. 167-171, 1965.

6. Utkin, V.I., "Quasi-Invariant Control of Forced Motion on Linear Plants with Variable Parameters," Engineering Cybernetics, No. 5, pp. 149-154, 1966.

7. Kwatny, H.G., Young, K.D., "Formulation and Dynamic Behaviour of a Variable Structure Servomechanism," proceedings of JACC, 1981.

8. Drazenovic, B., "The Invariance Conditions in Variable Structure Systems," Automatica, Vol. 5, pp. 287-295, 1969.

9. Young, P.C., Willems, J.C., "An Approach to the Linear Multivariable Servomechanism Problem," Int. J. of Control, Vol. 15, No. 5, pp. 961-979, 1972.

10. Popov, V.M., Hyperstability of Automatic Control Systems, Springer, New York, 1973.

11. Landau, Y.D., Adaptive Control-The Model Reference Approach, Marcel Decker, Inc., 1979.

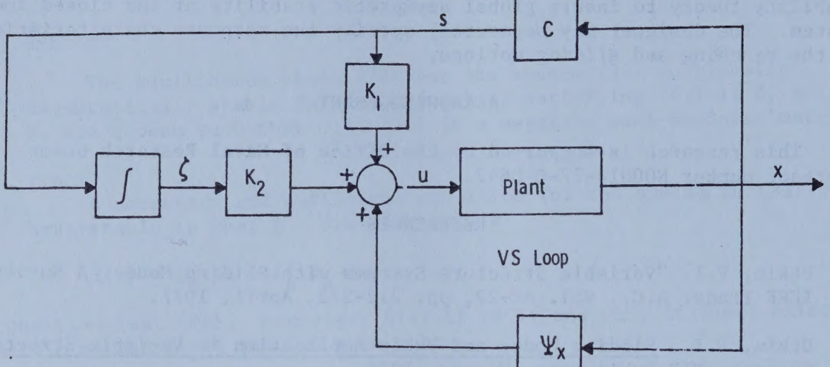

s
C
K₁
+
+
+
∫
ζ
K₂
u
Plant
x
VS Loop
Ψx

Figure 1 VS Servomechanism for Disturbance Rejection when $E_1 = 0$.

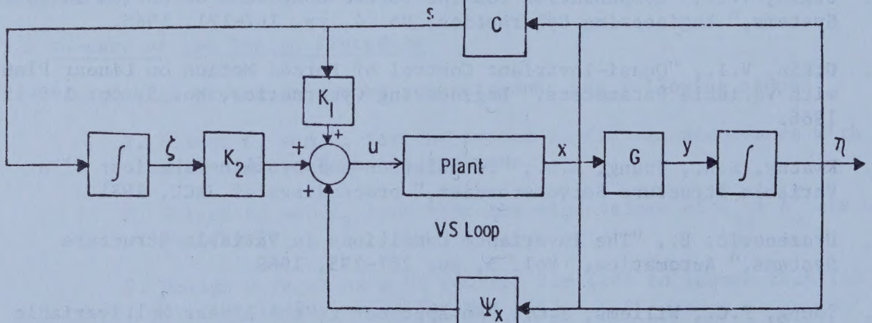

s
C
K₁
+
+
+
∫
ζ
K₂
u
Plant
x
G
y
∫
η
VS Loop
Ψx

Figure 2 VS Servomechanism for Output Regulation when $E_1 \neq 0$.

EIGENSTRUCTURE ASSIGNMENT WITH MINIMUM POLE SENSITIVITY TO PLANT PARAMETER VARIATIONS

SANG HYUK LEE*
Department of Electrical and Computer Engineering
University of Iowa
Iowa City, Iowa 52242

We consider here the pole assignment problem of linear multi-input time-invariant systems minimizing the sensitivity of assigned eigenvalues to the system parameter variations. As the state feedback matrix giving a specified closed-loop pole spectrum has some degree of freedom in the multi-input cases some methods of pole assignment minimizing the eigenvalue sensitivity were proposed in the past. In this paper, we first determine the subspaces of eigenvectors associated with the assigned eigenvalues in the augmented eigenvector space and then find out the unique eigenvectors in the subspace minimizing the pole sensitivity with respect to the global variation of system matrix. The corresponding feedback matrix is determined from this eigenstructure assignment.

For the parameter-perturbed system,

$$\dot{x} = (A+\delta A)x + Bu, \quad u = Kx$$

we determine the matrix K which assigns a given closed-loop pole spectrum and also minimizes the eigenvalue variations,

$$\delta x_i = v_i^T \delta A u_i, \quad i\varepsilon n \tag{1}$$

where u_i is the eigenvector associated with λ_i and v_i^T is the i-th row of the inverse eigenvector matrix.

From the eigenstructure relation,

$$[A-\lambda_i IB]\begin{bmatrix} u_i \\ q_i \end{bmatrix} = 0, \quad q_i = Ku_i,$$

the eigenvector satisfying the above equation can be given as

$$\begin{bmatrix} u_i \\ q_i \end{bmatrix} = \sum_{j=1}^{m} \alpha_i^j \begin{bmatrix} u_i^j \\ q_i^j \end{bmatrix}, \quad i\varepsilon n \tag{2}$$

We determine α_i^j from eqs. (1) and (2) minimizing eq. (1) and then the eigenstructure with minimum sensitivity will be uniquely determined. So, the desired feedback matrix is $K = QU^{-1}$. This eigenstructure assignment with minimum pole sensitivity is very simple and can be extended to the problems with feedback gain constraints or observer pole assignment.

---

*On leave from Ajou Institute of Technology, Suwon, Korea.

TRANSFORMATION OF NONHOMOGENEOUS NONLINEAR SYSTEMS

R. SU
National Research Council
& NASA Ames Research Center
Moffett Field, CA 94035

G. MEYER
NASA Ames Research
Center
Moffett Field, CA.

L. R. HUNT
Texas Tech. U.&
NASA Ames Research
Center, Moffett Field

ABSTRACT

It is pointed out that the origin of the state space is not essential to the theory previously developed by us. A necessary and sufficient condition for a nonhomogeneous nonlinear system to be equivalent to controllable linear systems is given.

SUMMARY

We consider systems of the form

$$\dot{x} = f(x,u) \tag{1}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}$, and the function f is smooth but does not necessarily vanish at (0,0). Let us consider transformations defined by state coordinate changes $y = y(x)$ and feedback $v = v(x,u)$. We assume that $\partial y/\partial x$ and $\partial v/\partial u$ are nonsingular and $y(0)$ and $v(0,0)$ need not vanish. These transformations actually establish an equivalence relation for systems (1).

A state x_o is called an equilibrium point if there is a control u_o such that $f(x_o,u_o) = 0$. We shall show that the transformations preserve the equilibrium points of a system, and give a necessary and sufficient condition for a system to be a linear equivalent near an equilibrium point.

Suppose two systems $\dot{x} = f(x,u)$ and $\dot{y} = g(y,v)$ are equivalent, then we have

$$\frac{\partial y}{\partial x} f(x,u) = g(y(x), v(x,u)).$$

Clearly, $f(x_o,u_o) = 0$ implies $g(y_o,v_o) = 0$ where $y_o = y(x_o)$ and $v_o = v(x_o,u_o)$.

Theorem A system (1) is equivalent to a controllable linear system if and only if

(i) $f(x,u) = f(x) + g(x) \cdot \phi(x,u)$ where $\phi(x,u)$ is a scalar function with $\partial\phi/\partial u \neq 0$,
(ii) the vectors g, $(ad^1 f,g)$, $\cdots$, $(ad^{n-1} f,g)$ span $\mathbb{R}^n$,
(iii) the set of vector fields g, $(ad^1 f,g)$, $\cdots$, $(ad^{n-2} f,g)$ is involutive.

The proof of this theorem in the homogeneous case given in [1] also works here. The idea is really independent of the homogeneity assumption. For the construction of a desired transformation, see [3]. Design schemes utilizing this theory can be found in [2].

REFERENCES

[1] R. Su, On the linear equivalents of nonlinear systems, submitted to Systems and Control Letters.
[2] G. Meyer, The design of exact nonlinear modal followers, Proceedings of the 1981 Joint Automatic Control Conference.
[3] L. R. Hunt and R. Su, Global transformations of nonlinear systems, Proceedings of the 1981 Joint Automatic Control Conference.
[4] R. W. Brockett, Feedback invariants ofr nonlinear systems, IFAC, 1978.

SOME RESULTS ON STABILITY AND STABILIZATION OF SYSTEMS WITH RETARDATION

ARILD THOWSEN
Department of Electrical Engineering
Iowa State University, Ames, Iowa 50011

ABSTRACT

The paper presents results on two topics: (1) stabilization of time-delay systems by memoryless feedback, and (2) conditions for stability of time-delay systems independent of the delay.

1. INTRODUCTION

Determination of the stability properties of a linear time-invariant differential-difference system is a difficult task because of the absence of analytic stability criteria which lend themselves to easy verification. The main stability theorem for finite-dimensional linear time-invariant systems, namely the system is asymptotically stable if and only if all the roots of the characteristic equation have negative real parts, also holds for the class of infinite dimensional systems governed by the differential equation with retarded arguments [1]

$$\dot{x}(t) = \sum_{i=0}^{m} A_i x(t-ih), \; h > 0, \; x(t) \varepsilon R^n \tag{1}$$

The characteristic function for (1) is a quasipolynomial or exponential polynomial which can be written as [2]

$$\Delta(\lambda, e^{-\lambda h}) = \lambda^n + \alpha_{n-1}(e^{-\lambda h})\lambda^{n-1} + \ldots + \alpha_0(e^{-\lambda h}) \tag{2}$$

where $\alpha_j(e^{-\lambda h})$, $j = 0, 1, \ldots, n-1$, are polynomials in $e^{-\lambda h}$. Only in rare cases is the determination of the zeroes of (2) straightforward. This occurs when system (1) has a finite spectrum so the characteristic function (2) reduces to a polynomial in λ only. Then the dynamic behavior of (1) will be governed by an ordinary differential equation (without retardation) after a time equal to a fixed multiple of the delay h [3]. In all other cases the characteristic equation for system (1) is a transcendental equation for which necessary and sufficient conditions for all the roots to lie in the open left half plane have been determined only in a few special cases [1,4,5]. Indirect methods for stability analysis have

---

This work was supported by the Engineering Research Institute of Iowa State University.

therefore enhanced appeal. The second method of Liapunov has been applied to derive stability conditions for (1) and feedback stabilizability conditions for the inhomogeneous system

$$\dot{x}(t) = Ax(t) + Bx(t-h) + Cu(t), \quad t > 0 \tag{3}$$

Here $x(t) \varepsilon R^n$, $u(t) \varepsilon R^m$, $1 \leq m \leq n$, and $h > 0$. Different forms of feedback stabilization of (3) may be considered depending on the information about the system state used by the feedback controller. The forms include state feedback with $u(t) = \int_{-h}^{0} \eta(\theta)x(t+\theta)d\theta$ [13]-[15], discrete-time feedback with $u(t) = \sum_{i=0}^{m} C_i x(t-\Delta i)$ (cfr. [23]); sampled-data feedback with $u(t) = \sum_{i=0}^{m} K_i x((r-i)\Delta) \ \forall \ t \varepsilon [r\Delta,(r+1)\Delta)$ [16]; and memoryless feedback [17]-[19].

Earlier conditions for memoryless feedback stabilization have been obtained via Liapunov's second method. In section 2 of this paper, new sufficiency conditions are derived using comparison arguments similar to those in [24].

An interesting result concerning stabilization of the linear time-varying system

$$\dot{x}(t) = A(t)x(t) + Bx(t-h(t)) + C(t)u(t) \tag{4}$$

was given in [20]. Using a property of the matrix solution to the differential Riccati-equation, the authors were able to show that with time-dependent feedback of the form $u(t) = K(t)x(t)$ the closed-loop system could be stabilized with any prescribed stability degree. In section 3 we present a time-invariant counterpart of their result with the feedback matrix K restricted to be time-independent. The result is easily proven using only elementary knowledge of the Liapunov equation.

Most sufficiency conditions for asymptotic stability of (1) are believed to be quite conservative. Frequently the overly restrictive conditions result from the choice of Liapunov functions which generally make the conditions obtained valid for all positive values of the delay h in (1). Stability concepts stronger than asymptotic stability have been considered, partly for their potential applicational value and partly in the hope that these alternative stability concepts correspond to system conditions which can be more readily tested than those for asymptotic stability for a fixed delay h. Two such stability concepts are asymptotic stability independent of delay and exponential asymptotic stability independent of delay. They have both been the topic of several recent papers [6-11]. Conditions which are both necessary and sufficient for these types of stability are now available in different forms. A result based on arguments presented in [8] is given in section 4. The new conditions are often simpler to check than previous results. The need to distinguish clearly between delay-independent asymptotic stability and delay-independent exponential asymptotic stability is illustrated. There is currently some imprecision in the literature concerning this point; it is probably rooted in the fact that for a fixed h (or even for a bounded range of h) a linear time-invariant system with retardation is asymptotically stable if and only if the system is exponentially asymptotically stable [12].

2. STABILIZATION

Given system (3) with feedback $u(t) = Kx(t)$. Assume there exists a matrix M of dimension n×n such that the pair (A+BM, C) is controllable. Then, for any $\beta > 0$, the exponential matrix $\phi(t) \triangleq \exp\{(A+BN+CK)t\}$ satisfies $\|\phi(t)\| \le ke^{-\beta t}$ for some finite $k \ge 1$ by proper choice of K. The dynamics of the closed-loop system can be written as

$$\dot{x}(t) = (A+BM+CK)x(t) - BM[x(t) - x(t-h)] + B(I-M)x(t-h) \qquad (5)$$

Define the expressions

$$g(t,Z(\cdot)) \triangleq \|BM\| \int_{t-h}^{t} [\,\|A+CK\|Z(s) + \|B\|Z(s-h)]ds + \|B(I-M)\|Z(t-h)$$

$$T(\alpha) \triangleq (h\|BM\| \; \|A+CK\| + h\|BM\| \; \|B\|e^{\alpha h} + \|B(I-M)\|)e^{\alpha h}$$

By integration of (5) and taking the norm we obtain

$$\|x(t)\| \le ke^{-\beta t}\|x(0)\| + k \int_{0}^{t} e^{-\beta(t-\tau)}g(\tau,x(\cdot))d\tau$$

The norm of x will be compared with the function v defined for any $\alpha > 0$ by

$$v(t) = \begin{cases} k|x_0|e^{-\alpha t}, & t \ge 0 \\ k|x_0|, & t < 0 \end{cases}$$

with $|x_0| \triangleq \sup\limits_{-2h\le s\le 0} \|x(s)\|$. It is easy to verify that $g(t,v(\cdot)) \le T(\alpha)v(t)$.

Since

$$\frac{dv(t)}{dt} + \beta v(t) - kg(t,v(\cdot)) \ge (\beta-\alpha-kT(\alpha))v(t),$$

integration under the assumption that $\beta - \alpha - kT(\alpha) > 0$ yields $v(t) > ke^{-\beta t}|x_0| + k \int_{0}^{t} e^{-\beta(t-\tau)}g(\tau,v(\cdot)d\tau$. It can then be shown that $\|x(t)\| \le v(t)$ for all $t \ge 0$. Hence, we have:

Theorem 1

If there exists a matrix M such that (A+BM, C) is controllable, system (3) with feedback $u(t) = Kx(t)$ is stabilizable with stability degree $\alpha > 0$ whenever $\beta - \alpha - kT(\alpha) > 0$.

Theorem 2

If (A,C) is controllable and $\beta > \alpha + ke^{\alpha h}||B||$, system (3) with feedback $u(t) = Kx(t)$ is stabilizable with stability degree $\alpha > 0$.

3. STABILIZATION WITH ARBITRARY STABILITY DEGREE

An interesting research problem is whether system (3) can be stabilized with any prespecified stability degree α using time-invariant memoryless feedback. This problem was investigated under the additional condition that the columns of B are linear combinations of the columns of C. (This set of conditions also appears in [21]-[22].) The next theorem provides an affirmative answer.

Theorem 3

If (A,C) is controllable and the columns of B are linear combinations of the columns of C, for any stability degree $\alpha > 0$ and any finite delay $h > 0$, there exists a time-invariant, memoryless feedback control $u(t) = Kx(t)$ for (1) such that $||x(t)|| \leq k|x_0|e^{-\alpha t}$, $t \geq 0$.

An elementary proof of Theorem 3, based on solution properties of the Liapunov equation, can be found in [25]. It should be noted that the feedback gain K in the theorem depends on both α and h. Hence, the theorem does not apply to study of stability properties independent of the delay [6]-[11], [26].

4. DELAY-INDEPENDENT STABILITY

In [8] it was shown how the Routh-Hurwitz criterion could be used to investigate asymptotic stability of simple linear delay systems for all values of $h > 0$. The method was based on the idea that the system should be asymptotically stable for $h = 0$ and that the characteristic equation for (1) should have no roots on the imaginary axis for any value of $h > 0$. The same arguments are used to derive a necessary and sufficient condition for delay-independent asymptotic stability of system (1). A sufficient condition for the stronger property of delay-independent exponential asymptotic stability is also obtained. Kamen [7] has shown by a different method that this last condition is also necessary. Since the range of h is unbounded, it is necessary to distinguish between the following stability concepts:

Definition 1

System (1) is asymptotically stable if for all initial functions $\phi \varepsilon C([-mh,0];R^n)$ the solutions $x(t,\phi)$ satisfy $||x(t,\phi)|| \to 0$ as $t \to \infty$.

Definition 2 [12]

System (1) is exponentially asymptotically stable if for all initial functions $\phi \varepsilon C([-mh,0];R^n)$ there exist constants $\alpha > 0$, $\beta(\phi) > 1$ such that the solution $x(t,\phi)$ of (1) satisfies $||x(t,\phi)|| \leq \beta(\phi)e^{-\alpha t}$ for all $t \geq 0$.

Delay-independent exponential asymptotic stability is a stronger property than delay-independent asymptotic stability. In addition to requiring that the system has all its poles in the open left half plane (LHP), exponential asymptotic stability independent of delay also requires that there are no poles arbitrarily close to the imaginary axis for any $h \geq 0$.

Based upon the reasoning presented in [8], we can prove (cfr. [26]).

Theorem 4

Necessary and sufficient conditions for delay-independent asymptotic stability of system (1) are that

$$\mathrm{Re}\lambda_i\left(\sum_{i=0}^{m} A_k \right) < 0, \quad i = 1,2,\ldots,n$$

and

$$\left| j\omega I - \sum_{k=0}^{m} A_k e^{j\theta k} \right| \neq 0, \quad \forall\ \omega \neq 0, \quad \forall\ \theta \varepsilon [0,\pi]$$

Theorem 5

A sufficient condition for delay-independent exponential asymptotic stability of (1) is that

$$\mathrm{Re}\lambda_i\left(\sum_{k=0}^{m} A_k e^{j\theta k} \right) < 0, \quad \forall\ \theta \varepsilon [0,\pi], \quad i = 1,2,\ldots,n.$$

Using certain identities for polynomials in two variables [6], Kamen [7] has shown that the condition in Theorem 2 is also necessary for exponential asymptotic stability independent of delay.

Finally we illustrate by an example the difference between asymptotic and exponential asymptotic stability. The system

$$\dot{x}(t) = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} x(t) + \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix} x(t-h)$$

is asymptotically stable independent of delay (cfr. Theorem 4) but not exponentially asymptotically stable independent of delay.

6. SUMMARY

Sufficient conditions for memoryless feedback stabilization of linear time-invariant delay systems are derived in the first half of the paper. The second half deals with the determination of necessary and sufficient conditions for linear time-invariant systems with multiple delays to be asymptotically stable.

REFERENCES

1. R. E. Bellman and K. L. Cooke, Differential-Difference Equations, Academic Press, 1963.
2. F. Kappel, "Degenerate difference-differential equations: Algebraic theory", J. Differential Equations, Vol. 24, 1977, pp. 99-126.
3. D. Henry, "Small solutions of linear autonomous functional differential equations", J. Differential Equations, Vol. 8, 1970, pp. 494-501.
4. J. K. Hale, Theory of Functional Differential Equations, Springer-Verlar, 1977.
5. N. D. Hayes, "Roots of the transcendental equation associated with a certain difference-differential equation", J. London Math. Soc., Vol. 25, 1950, pp. 226-232.
6. E. W. Kamen, "On the relationship between zero criteria for two-variable polynomials and asymptotic stability of delay differential equations", IEEE Trans. Automat. Contr., Vol. AC-25, No. 5, 1980, pp. 983-984.
7. E. W. Kamen, "Linear systems with commensurate time delays: Stability and stabilization independent of delay", to appear in IEEE Trans. Automat. Contr.
8. A. Thowsen, "The Routh-Hurwitz method for stability determination of linear differential-difference systems", Int. J. Control, Vol. 33, No. 5, 1981, pp. 991-995.
9. R. M. Lewis and B. D. O. Anderson, "Necessary and sufficient conditions for delay-independent stability of linear autonomous systems", IEEE Trans. Automat. Contr., Vol. AC-25, No. 4, 1980, pp. 735-739.
10. S. D. Brierly, J. N. Chiasson, E. B. Lee and S. H. Zak, "On stability independent of delay for linear systems", Preprint.
11. Z. V. Rekasius, "Delay-independent stability of linear systems", Proc. 1981 Joint Automat. Contr. Conf., Charlottesville, VA, June 1981, paper WP-4E.
12. L. E. Elsgolts, Introduction to the Theory of Differential Equations with Deviating Arguments, Holden-Day, 1966.
13. Y. S. Osipov, "Stabilization of controlled systems with delays", Differentsial'nye Uravneniya, Vol. 1, No. 5, 1965, pp. 605-618.

14. A. Manitius and R. Triggiani, "Sufficient conditions for function space controllability and feedback stabilizability of linear retarded systems", Proc. 1976 IEEE Conf. on Decision and Control, Clearwater, Fla., pp. 1209-1216.

15. L. Pandolfi, "On feedback stabilization of functional differential equations", Bull. Un. Mat. Ital., Vol. II, No. 4, 1975, pp. 626-635.

16. A. Thowsen, "Stability of time-delay systems with sampled feedback", to appear in Int. J. Control.

17. W. H. Kwon and A. E. Pearson, "A note on feedback stabilization of a differential-difference system", IEEE Trans. Automat. Contr., Vol. AC-22, No. 3, 1977, pp. 468-470.

18. G. J. Nazaroff and G. A. Hewer, "Stabilization of linear, autonomous differential delay systems", IEEE Trans. Automat. Contr., Vol. AC-18, No. 6, 1973, pp. 673-674.

19. A. Feliachi and A. Thowsen, "Memoryless stabilization of linear delay-differential systems", IEEE Trans. Automat. Contr., Vol. AC-26, No. 2, 1981, pp. 586-587.

20. M. Ikeda and T. Ashida, "Stabilization of linear systems with time-varying delay", IEEE Trans. Automat. Contr., Vol. AC-24, No. 2, 1979, pp. 369-370.

21. G. J. Nazaroff, "Stability and stabilization of linear differential delay systems", IEEE Trans. Automat. Contr., Vol. AC-18, No. 3, 1973, pp. 317-318.

22. D. W. Ross and I. Flugge-Lotz, "An optimal control problem for systems with differential-difference equation dynamics", SIAM J. Control, Vol. 7, No. 4, 1969, pp. 609-623.

23. D. W. Ross, "Controller design for time-lag systems via a quadratic criterion", IEEE Trans. Automat. Contr., Vol. AC-16, No. 6, 1971, pp. 664-672.

24. V. Z. Tsalyuk, "An estimation of the magnitude of solutions of a system of linear differential equations with delay", Differentsial'nye Uravneniya, Vol. 9, No. 11, 1973, pp. 2099-2101.

25. A. Thowsen, "Stabilization of a class of linear time-delay systems", submitted.

26. A. Thowsen, "Delay-independent asymptotic stability of linear systems", submitted.

A Z-TRANSFORM THEORY FOR DISTRIBUTED SENSING AND CONTROL*

SHELDON S.L. CHANG
Department of Electrical Engineering
State University of New York
Stony Brook, N.Y. 11794

ABSTRACT

In linear systems theory each component of the state vector $\underline{\xi}(t)$ is a single function of time. In spatially dicretized distributed parameter theory each component of $\underline{\xi}(\underline{z},t)$ represents an array of infinitely many such functions. The system matrices $\underset{\sim}{A}, \underset{\sim}{B}, \underset{\sim}{C}$ and $\underset{\sim}{D}$ become $\underset{\sim}{A}(\underline{z},t)$, $\underset{\sim}{B}(\underline{z},t)$, $\underset{\sim}{C}(\underline{z},t)$ and $\underset{\sim}{D}(\underline{z},t)$. A square matrix $\underset{\sim}{M}(\underline{z})$ is said to be invertible if det $\underset{\sim}{M}(\underline{z})$ is non-zero on Z_c ($Z_c \stackrel{\Delta}{=} \{\underline{z}: |z_i| = 1, \; i = 1,2 \dots N.\}$). An n x k matrix $\underset{\sim}{F}(\underline{z})$ is said to be of generalized rank n if $\underset{\sim}{F}(\underline{z})\underset{\sim}{F}(\underline{z})^{\dagger}$ is invertible.

It is shown that with "invertible" replacing "non-singular", and "generalized rank" replacing "rank" practically every result of ordinary linear systems theory can be duplicated in the distributed parameter theory.

A criterion for discretization, and an application example is also included in the paper.

I. INTRODUCTION

The study of large distributed parameter systems have important applications in processing, nuclear, and aero-space industries and also environmental control. The systems are modeled by partial differential equations, and there is a substantial literature on the identification, estimation, and control of these systems. [1], [2]. The work largely parallel ordinary systems theory with the introduction of integral signs and kernels at appropriate places. The three aspects of the problem: theory computational algorithm, and realization of the actual system are generally treated in separate papers or reports.

The present paper introduces an alternative approach for a class of systems which can be approximated by spatial discretization: thermal conduction, elasticity and vibration, and certain chemical reactors etc. The theory leads explicitly to its realization by distributed (micro) processor control, including the required control and estimation algorithm. The engineering significance of this approach is its simplicity and the reliability of the resulting system.

The following parts of the theory will be presented concisely:

1. criteria of space discretization,
2. controllability and observability,
3. stability theorem,
4. optimum control and estimation, and an application example.

---

* Work performed under National Science Foundation Grant ECS-8007687.

Notations

Let $\underline{x}$ denote the space coordinates $(x_1, x_2 \ldots . x_N)$, and $\bar{f}(\underline{x})$ denote a function $\bar{f}$ of $x_1, x_2, \ldots . x_N$. The z-transform of $\bar{f}(\underline{x})$ is represented as $f(\underline{z})$:

$$f(\underline{z}) = \sum_{\underline{n}=-\infty}^{\underline{n}=\infty} \bar{f}(\underline{n}\underset{\sim}{L})\underline{z}^{-\underline{n}} \tag{1}$$

where $\displaystyle\sum_{\underline{n}=-\infty}^{\underline{n}=\infty} \triangleq \sum_{n_1=-\infty}^{n_1=\infty} \; \sum_{n_2=-\infty}^{n_2=\infty} \cdots \sum_{n_N=-\infty}^{n_N=\infty}$, $\underline{z}^{-\underline{n}} \triangleq z_1^{-n_1} z_2^{-n_2} \cdots z_N^{-n_N}$,

and $\underline{n}\underset{\sim}{L}$ is the vector $(n_1L_1, n_2L_2, \ldots . n_NL_N)$.

(Note: We use the overhead bar to differentiate the real spatial function $\bar{f}(\underline{x})$ from the transformed $f(\underline{z})$ as the latter is used more frequently in the present paper.) We denote the 2N-dimensional z-space as Z., and the unit circle product space as Z_c (See ABSTRACT). The symbol † denotes replacing each z_i by z_i^{-1}, i=1, 2 .... N, and transposition of matrix elements: $\underset{\sim}{M}(\underline{z})^{\dagger} = \underset{\sim}{M}(\underline{z}^{-1})^T$. Thus for $\underline{z}$ on Z_c, and any rational matrix function $\underset{\sim}{M}(\underline{z})$ with real coefficients.

$$\underset{\sim}{M}(\underline{z})^{\dagger} = \underset{\sim}{M}(\underline{z})^{*} \tag{2}$$

where * denotes Hermitian conjugation.

Definitions

The function $f(\underline{z})$ is said to be (exponentially) convergent if $\bar{f}(\underline{x}) \rightarrow 0$ (exponentially) as $|x| \rightarrow \infty$. A vector function is convergent if its every component is convergent.

The definitions of "invertible" and "generalized rank" are stated in the ABSTRACT. They follow from Theorem 1:

Theorem 1 If a polynomial $P(\underline{z})$ is non-zero on Z_c, its reciprocal $1/P(\underline{z})$ has a unique, exponentially convergent expansion. Thus if det $\underset{\sim}{M}(\underline{z})$ is non-zero on Z_c, $\underset{\sim}{M}(\underline{z})^{-1}$ has a unique expansion. Theorem 1 is proved in [3] which also gives an algorithm for the expansion.

II. CRITERIA OF DISCRETIZATION

Multi-dimensional sampling theorem

Let $\bar{f}(\underline{x})$ denote a band-limited signal with spatial frequency limit Ω_i along x_i. Then $\bar{f}(\underline{x})$ can be expressed as

$$\bar{f}(\underline{x}) = \sum_{\underline{n}=-\infty}^{\underline{n}=\infty} \bar{f}(\underline{n}\underset{\sim}{L})h(\underline{x}-\underline{n}\underset{\sim}{L}) \tag{3}$$

where $\underset{\sim}{L}$ is a diagonal matrix with diagonal elements $L_i = \pi/\Omega_i$; and $h(\underline{x})$ is an interpolation function:

$$h(\underline{x}) = \prod_{i=1}^{i=N} \left(\frac{\sin \Omega_i x_i}{\Omega_i x_i} \right) \tag{4}$$

The proof of the theorem is similar to that of its one-dimensional version.

Equation (3) means that the sampling points form an N-dimensional lattice with spacings L_i, i=1,2 .... N, and the value of $\overline{f}(\underline{x})$ everywhere is completely determined by its values at the lattice points. The sampling theorem is a justification for space or space-time discretization of bandlimited signals. In practice the energies taper off at high frequencies, and the justification is in an approximate sense.

Natural Justification for Space Discretization in Practical Applications

Some typical examples of practical applications cited in the literature are listed below:

(a) Thermal conduction

$$C \frac{\partial v}{\partial t} = K \nabla^2 v + q \tag{5}$$

(b) Diffusion and chemical reaction

$$\frac{\partial v}{\partial t} + \nabla.(vV) = D \nabla^2 v - av \tag{6}$$

(c) Elastic vibrations

$$\frac{\partial v}{\partial t} = w \tag{7}$$

$$\mu \frac{\partial w}{\partial t} + \delta_E . w + \delta_I \frac{\partial^4 w}{\partial x^4} - \frac{\partial}{\partial x}\left(T \frac{\partial v}{\partial x}\right) + \frac{\partial^2}{\partial x^2}\left(\alpha \frac{\partial^2 v}{\partial x^2}\right) = p(x) \tag{8}$$

In (5), (6), and (7) the state variable v represents temperature, concentration, and transversal displacement respectively. The vector V in (6) represents solvent fluid velocity, and a is a reactivity coefficient. In (7), μ is the mass per unit length, δ_E and δ_I are the external and internal damping coefficients respectively, T is the tensil force and α is a flexibility coefficent for beams. The coefficient α is zero for vibrating strings and membranes.

Let $\omega = (\omega_1, \omega_2, ...)$ represents the spatial frequencies, and ω is the magnitude of $\underline{\omega}$. Applying spatial Fourier transform to (5) to (8) yields four ordinary differential equations in t. Examination of these equations show that they have the following exponentially decaying constants:

(a) $K\omega^2/C$, (b) $a + D\omega^2$, and (c) $(\delta_E+\delta_I \omega^4)/2\mu$.

Thus the system partial differential equations impose a natural spatial frequency limit: Fourier components with large ω are quickly damped out.

Effectiveness of Spatially Discrete Control

We assume that spatially discrete controls $\bar{u}(\underline{n}\underset{\sim}{L})$ are applied at lattice points through diffusers $h(\underline{x}-\underline{n}\underset{\sim}{L})$:

$$\underline{\bar{u}}(\underline{x}) = \sum_{\underline{n}} \underline{\bar{u}}(\underline{n}L)h(\underline{x}-\underline{n}L) \tag{9}$$

The diffusers are used to avoid exciting high spatial frequency modes. Similarly, measurements of the state variables $\bar{\underline{\xi}}(\underline{n}L)$ at the lattice points are also through diffusers or spatially averaged to avoid aliasing. Under these assumptions, spatially discrete controls have no effect on high spatial frequency Fourier components ($\omega_i > \pi/L_i$, any i).

Example 1 The equation for temperature control of a one dimensional slab can be expressed as

$$C\frac{\partial v}{\partial t} = K\frac{\partial^2 v}{\partial x^2} + u(x,t) + w(x,t) \tag{10}$$

and w is an external stochastic disturbance:

$$<w(x,t)\ w(x',t')> = W\delta(x-x',\ t-t') \tag{11}$$

Eq. (11) represents w as white in both time and space. It is the worst case possible. With feedback stabilization, the temperature fluctuations with spatial frequencies below π/L is negligible. Determine the mean square value of the remaining fluctuation under steady state condition.

Solution For $\omega > \pi/L$, the spectral density of v is

$$\Phi_v(\omega_o,\ \omega) = \frac{W}{\omega_o^2C^2 + \omega^4K^2} \tag{12}$$

The mean square fluctuation is

$$\overline{v^2} = \frac{1}{2\pi^2}\int_{-\infty}^{\infty}\int_{\pi/L}^{\infty} \Phi_v(\omega_o,\omega)\ d\omega_o d\omega = \frac{WL}{2\pi^2 CK} \tag{13}$$

The above example shows that with completely random disturbance, the mean square value of the irreducible temperature fluctuation is proportional to the regulator spacing L. Conversely L can be determined by the allowed temperature fluctuations.

III. CONTROLLABILITY AND OBSERVABILITY

A linear distributed parameter system with space discretization can be represented as

$$\frac{\partial\underline{\xi}(\underline{z},t)}{\partial t} = \underset{\sim}{A}(\underline{z},\ t)\ \underline{\xi}(\underline{z},t) + \underset{\sim}{B}(\underline{z},t)\underline{u}(\underline{z},t) \tag{14}$$

$$\underline{\eta}(\underline{z},t) = \underset{\sim}{C}(\underline{z},t)\underline{\xi}(\underline{z},t) + \underset{\sim}{D}(\underline{z},t)\underline{u}(\underline{z},t) \tag{15}$$

where $\underline{z}$ is an N-dimensional transform variable; $\underline{\xi}$,$\underline{u}$, and $\underline{\eta}$ are nxl state vector, mxl control vector, and m'xl observed or output vector respectively.

In a system with both space and time discretization, (14) is replaced by

$$\underline{\xi}(\underline{z},t+1) = \underset{\sim}{A}(\underline{z},t)\underline{\xi}(\underline{z},t) + \underset{\sim}{B}(\underline{z},t)\underline{u}(\underline{z},t) \tag{16}$$

Definitions

A distributed system is controllable at t if given any convergent target state $\underline{\xi}_d(\underline{z})$, there exist a time $t_1>t$, and convergent $\underline{u}(\underline{z},\cdot)$ which brings the system from an initial state $\underline{\xi}_o(\underline{z})$ to $\underline{\xi}_d(\underline{z})$:

$$\underline{\xi}(\underline{z},t_1) = \underline{\xi}_d(\underline{z})$$

A distributed system is observable at t if given any convergent initial state $\underline{\xi}(\underline{z},t)$, there exists a $t_1>t$ such that $\underline{\xi}(\underline{z},t)$ is completely determinable from the observed $\underline{\eta}(\underline{z},\cdot)$ in the interval $[t,t_1]$.

Theorem 2 Let $\underset{\sim}{f}(\underline{z},t)$ be an nxm matrix function defined on ZxT, where T contains the interval $[t_o, t_1]$ as its interior points; and $\underset{\sim}{f}(\underline{z},t)$ is analytic in t. Let $\underset{\sim}{E}(\underline{z},t)$ denote the nx(nxm) matrix:

$$\underset{\sim}{E}(\underline{z},t) = \{\underset{\sim}{f}(\underline{z},t) \,\vdots\, \underset{\sim}{f}^{(1)}(\underline{z},t) \,\vdots\, \cdots\cdots \,\vdots\, \underset{\sim}{f}^{(n-1)}(\underline{z},t)\}$$

where $\underset{\sim}{f}^{(k)}(\underline{z},t) = \dfrac{\partial^k \underset{\sim}{f}(\underline{z},t)}{\partial t^k}$. Let $\underset{\sim}{W}(\underline{z})$ be a matrix function of $\underline{z}$ defined by

$$\underset{\sim}{W}(\underline{z}) = \int_{t_o}^{t_1} f(\underline{z},t)f(\underline{z},t)^{\dagger}dt \tag{17}$$

If $E(\underline{z},t_a)$ is of generalized rank n for some t_a in the interval $[t_o,t_1]$, then $\underset{\sim}{W}(\underline{z})$ is invertible.

Proof The above theorem will be proved by contradiction. Assuming that $\underset{\sim}{W}(\underline{z})$ is not invertible, then for some $\underline{z}_a$ on Z_c

$$\det.\ \underset{\sim}{W}(\underline{z}_a) = 0 \tag{18}$$

From (2) $\underset{\sim}{W}(\underline{z})$ as defined by (17) is a Hermitian matrix. Eq. (18) implies the existence of a constant vector $\underline{\alpha}$ such that

$$\underline{\alpha}\ \underset{\sim}{W}(\underline{z}_a)\alpha^* = 0 \tag{19}$$

From (17), (19) can be rewritten as

$$\int_{t_o}^{t_1} \underline{\alpha}\underset{\sim}{f}(\underline{z}_a,t)\ \underset{\sim}{f}(\underline{z}_a,t)^*\underline{\alpha}^*dt = 0 \tag{20}$$

Since the integrand is non-negative, (20) requires that

$$\underline{\alpha}\underset{\sim}{f}(\underline{z}_a,t) = 0 \qquad t_o \leq t \leq t_1 \tag{21}$$

Repeated differentiation of (21) gives

$$\underline{\alpha}\underset{\sim}{f}^{(k)}(\underline{z}_a,t) = 0 \qquad t_o \leq t \leq t_1 \tag{22}$$

Eq. (22) contradicts the given condition that $\underset{\sim}{E}(\underline{z},t_a)$ is of generalized rank n.

Theorem 3 The control system (16) is controllable at t if for some k the matrix $F(\underline{z},t)$ defined by

$$\underset{\sim}{F}(\underline{z},t) \triangleq (\underset{\sim}{B}(\underline{z},t+k) \,\vdots\, \underset{\sim}{A}(\underline{z},t+k)\underset{\sim}{B}(\underline{z},t+k-1) \,\vdots\, \ldots \,\vdots\, \underset{\sim}{A}(\underline{z},t+k)..\underset{\sim}{A}(\underline{z},t+1)B(\underline{z},T)) \tag{23}$$

is of generalized rank n.

Proof Let $\underline{\xi}_o(t+k+1)$ and $\underline{\mathcal{U}}(\underline{z})$ be defined as

$$\underline{\xi}_o(\underline{z},t+k+1) = A(\underline{z},t+k)A(\underline{z},t+k-1) \ldots A(\underline{z},t)\underline{\xi}_o(\underline{z}) \tag{24}$$

$$\underline{\mathcal{U}}^T(\underline{z}) = (\underline{u}(\underline{z},t+k)^T, \underline{u}(\underline{z},t+k-1)^T, \ldots \underline{u}(\underline{z},t)^T) \tag{25}$$

Repeated application of (16) gives

$$\underline{\xi}(\underline{z},t+k+1) = \underline{\xi}_o(\underline{z},t+k+1) + \underset{\sim}{F}(\underline{z},t)\,\underline{\mathcal{U}}(\underline{z}) \tag{26}$$

Therefore

$$\underline{\mathcal{U}}(\underline{z}) = F(\underline{z},t)^{\dagger}\,[F(\underline{z},t)F(\underline{z},t)^{\dagger}]^{-1}[\underline{\xi}_d(\underline{z})-\underline{\xi}_o(\underline{z},t+k+1)] \tag{27}$$

brings $\underline{\xi}(\underline{z},t+k+1)$ to $\underline{\xi}_d(\underline{z})$.

Theorem 4 Let $\underset{\sim}{M}_k(\underline{z},t)$, K=0,1...N-1, be defined by

$$\underset{\sim}{M}_o(\underline{z},t) = \underset{\sim}{B}(\underline{z},t) \tag{28}$$

$$\underset{\sim}{M}_{k+1}(\underline{z},t) = -\underset{\sim}{A}(\underline{z},t)M_k(\underline{z},t) + \frac{\partial}{\partial t}\underset{\sim}{M}_k(\underline{z},t) \tag{29}$$

If a $t_a > t$ exists such that the matrix

$$\underset{\sim}{F}(\underline{z},t_a) \triangleq (\underset{\sim}{M}_o(\underline{z},t_a) \,\vdots\, \underset{\sim}{M}_1(\underline{z},t_a) \,\vdots\, \ldots \,\vdots\, \underset{\sim}{M}_{n-1}(\underline{z},t_a))$$

is of generalized rank n, then the control system defined by (14) is controllable.

Proof $\underset{\sim}{M}_k(\underline{z},t)$ is related to the transition matrix $\underset{\sim}{\Phi}(\underline{z};t,\sigma)$ by

$$\frac{d^k[\underset{\sim}{\Phi}(\underline{z};t,\sigma)\underset{\sim}{B}(\underline{z},\sigma)]}{d\sigma^k} = \underset{\sim}{\Phi}(\underline{z},t,\sigma)\underset{\sim}{M}_k(\underline{z},\sigma)$$

Let $t_1 > t_a$, and

$$\underset{\sim}{W}(\underline{z}) = \int_{t}^{t_1} [\underset{\sim}{\Phi}(\underline{z};t,\sigma)\underset{\sim}{B}(\underline{z},\sigma)][\underset{\sim}{\Phi}(\underline{z};t,\sigma)\underset{\sim}{B}(\underline{z},\sigma)]^{\dagger} d\sigma$$

From Theorem 2, $\underset{\sim}{W}(\underline{z})$ is invertible. The required $\underline{u}(\underline{z},\sigma)$, $t \leq \sigma \leq t_1$, is

$$\underline{u}(\underline{z},\sigma) = [\underset{\sim}{\Phi}(\underline{z};t,\sigma)\underset{\sim}{B}(\underline{z},\sigma)]^{\dagger} W(\underline{z})^{-1}[\underline{\xi}(\underline{z})-\underline{\xi}_o(\underline{z},t_1)] \tag{30}$$

Theorem 5. The dynamic system defined by (14) and (15) (or (16) and (15)) is observable if $[A(z,t)^T, C(z,t)^T)]$ is controllable.

IV. STABILITY OF TIME-INVARIANT SYSTEMS

The matrices A, B, C, and D in (14) to (16) are independent of t. We refer to the time-invariant equations as (14A) to (16A) respectively.

Definition

A system is said to be stable if given any convergent initial state $\underline{\xi}_o(\underline{z})$, $\underline{\xi}(\underline{z},t)$ approaches zero as t approaches infinity.

Theorem 6 The system described by (14A) with $\underline{u}(\underline{z},\underline{t}) = 0$ is stable if det. $\{s\underset{\sim}{I}-\underset{\sim}{A}(\underline{z})\}$ is non-zero on the product space $\overline{Z}_d x \overline{R}$, where $\overline{R}$ is the right half of the s-plane including the imaginary axis.

Theorem 7 The system described by (16A) with $\underline{u}(\underline{z},t)=0$ is stable if det. $\{\underline{z}_o\underset{\sim}{I}-\underset{\sim}{A}(\underline{z})\}$ is non-zero on the product space $Z_c \times \overline{E}$ where $\overline{E} = \{\underline{z}_o \; \underline{z}_o \geq 1\}$

Theorems 6 and 7 are proved in [4]. The following theorem is an extension of ordinary linear systems eigenvalue assignment theorem, and can be similarly proved.

Theorem 8 Let $[\underset{\sim}{A}(\underline{z}), \underset{\sim}{B}(\underline{z})]$ be controllable. The eigenvalues of $\underset{\sim}{A}(\underline{z}) + \underset{\sim}{B}(\underline{z})K(\underline{z})$ can be arbitrarily assigned for every $\underline{z}$ on Z_c.

The above theorem means that a controllable distributed system can be stabilized by linear state feedback. Similarly, a stable generalized Luenberger observer can be found if the distributed system is observable.

V. OPTIMUM FILTERING AND CONTROL

The problem formulation and results on optimum filtering and control are completely identical with that of ordinary linear systems theory. The white noise processes are replaced by processes which are temporally white but spatially correlated. The same Kalman equations are obtained with all matrices replaced by matrix functions of $\underline{z}$.

Example 2 The one dimensional temperature control problem (10), (11) is spatially discretized to give:

$$C \frac{\partial v(z,t)}{\partial t} = \frac{K}{L^2}(z-2+z^{-1})v(z,t) + u(z,t) + w(z,t) \tag{31}$$

$$\Phi_{ww}(z) = WL \qquad (32)$$

The fluctuations in u and v under steady state operation are to be reduced by minimizing F:

$$F = P < v^2 > + Q < u^2 > \qquad (33)$$

Dynamic programming shows that the minimum value for F is

$$F = \frac{1}{2\pi j C^2} \oint Tr[R(\underline{z})\ \Phi_{ww}(z.t)] \frac{dz}{z} \qquad (34)$$

where R(z) satisfies the Riccati equation:

$$-\frac{\partial R}{\partial t} = AR + RA' + P - RBQ^{-1}B'R = 0 \qquad (35)$$

Solving (35) for R(z) gives

$$R(z) = \frac{L^2CP}{K\ k^2} [z-2+z^{-1} + \sqrt{(z-2+z^{-1})^2 + k^2}] \qquad (36)$$

where

$$k^2 = L^4P/K^2Q \qquad (37)$$

Let $J_o(k)$ be defined as

$$J_o(k) \triangleq \frac{1}{2\pi j k^2} \oint [z-2+z^{-1} + \sqrt{(z-2+z^{-1})^2 + k^2}] \frac{dz}{z}$$

$$= -\frac{2}{k^2} + \frac{1}{2\pi k} \int_0^{2\pi} (1 + \frac{16}{k^2} \sin^4 \frac{\theta}{2})^{1/2} d\theta \qquad (38)$$

From (34)

$$F = \frac{WL^3P}{K} J_o(k) \qquad (39)$$

The optimum control law is given by

$$u(z) = -\frac{1}{CQ} R(z)v(z) \qquad (40)$$

The function R(z) has a unique convergent expansion in z. Thus the control law requires inputs from everywhere.

Example 3 The problem of Exampoe 2 is modified by restricting the control law to

$$u(z) = -K_1 v(z) \tag{41}$$

where K_1 is a constant. Under steady-state condition, the spectral density function of v is

$$\Phi_v(z, \omega_o) = \frac{WL}{C^2\omega_o^2 + [\frac{K}{L^2}(z-2+z^{-1}) - K_1]^2} \tag{42}$$

and (33) becomes

$$F = (P+QK_1^2)\frac{1}{4\pi^2 j}\oint\int_{-\infty}^{\infty}\Phi_v(z,\omega_o)d\omega_o \frac{dz}{z} \tag{43}$$

Integrating first with respect to ω_o and then with respect to z gives

$$F = \frac{WL(P+QK_1^2)}{2C\sqrt{K_1(K_1+4K/L^2)}} \tag{44}$$

Differentiating F with respect to K_1 and setting the result to zero gives

$$K_1 = \frac{K}{L^2}\ a(k) \tag{45}$$

where the parameter k is defined by (37), and a(k) is related to k by

$$k = a\sqrt{\frac{a+6}{a+2}} \tag{46}$$

Note that the right-hand side of (46) increases monotonously with a and its inverse function a(k) is uniquely specified.

Substituting (45) and (46) into (44) and simplifying the result gives

$$F = \frac{WL^3P}{K}\ J_1(k) \tag{47}$$

where

$$J_1(k) = \frac{\sqrt{a(a+4)}}{a(a+6)} \tag{48}$$

The functions $J_o(k)$ and $J_1(k)$ are plotted in Fig. 1. We note that $J_1(k)$ is somewhat larger than $J_o(k)$ for all values of k.

The above two examples illustrate an important disadvantage of optimum control that the control law at every location depends on the feedback signals from all other locations. To realize such an algorithm a central processor with a communication network to all sensors and actuators become necessary. A better approach is to restrict the control law to depend only on local feedback signals, that is, $\underset{\sim}{K}_1(\underline{z})$ is a polynomial in $\underline{z}$. While the cost integral has a somewhat higher value, distributed processor control can be used to reduce the cost of implementation and to improve system reliability.

REFERENCES

[1] Control Theory of Systems Governed by Partial Differential Equations. A.K. Azia, J.W. Wingate, and M.J. Balas, Editors, Academic Press, Inc. New York, 1977.

[2] Distributed Parameter Systems, W.H. Ray and D.G. Lainiotis, Editors, Marcel Dekker, Inc. New York, 1978.

[3] S.S.L. Chang, "Digital Linear Processor Theory and Optimum Multidemensional Data Estimation", IEEE Trans. on Automatic Control, Vol. AC-24, 190-201, April 1979.

[4] S.S.L. Chang, "Temporal Stability of N-Dimensional Linear Processors and Its Applications", IEEE Trans. on Circuits and Systems, August, 1980.

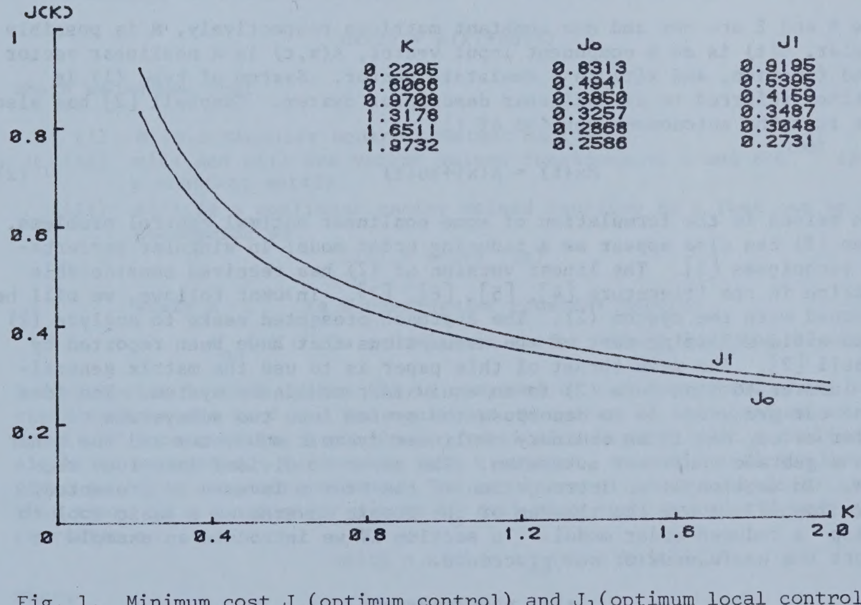


Fig. 1. Minimum cost J_o(optimum control) and J_1(optimum local control) as functions of the parameter k. (Computed by Wen-min Pan, Visiting Scholar from PRC.)

AN EFFICIENT METHOD FOR ANALYZING A CLASS OF NONLINEAR SEMISTATE EQUATIONS

M. SAIDAHMED and M. ELWAKKAD ZAGHLOUL
School of Engineering and Applied Science
Department of Electrical Engineering and Computer Science
The George Washington University
Washington, D.C. 20052

ABSTRACT

In this paper an efficient method for analyzing a class of nonlinear semistate equations that arise in circuit and control is described. We will be concerned with the problem of the form $M\dot{x}(t) = A(x) + Bu(t)$, where $x(t)$, $u(t)$ are vector valued functions, M is a singular constant matrix, and $A(x)$ is a nonlinear vector valued function. System of this form is sometimes referred to as nonlinear discriptor system. The approach is based on the concept of matrix generalized inverses, especially the Dr azin inverse, on which a porcedure for obtaining a reduced-order system is obtained. An illustrative example is given.

I. INTRODUCTION

This paper is concerned with analyzing a class of nonlinear semistate equations. Recently, Newcomb [1] has shown that some systems of linked differential and algebraic equations arising in circuit theory not possessing a state variable representation may admit semistate equations of the form

$$M\dot{x}(t) = A(x,t)+Bu(t) \tag{1}$$

where M and B are nxn and nxm constant matrices respectively, M is possibly singular, $u(t)$ is an m component input vector, $A(x,t)$ is a nonlinear vector valued function, and $x(t)$ is a semistate vector. System of type (1) is sometimes referred to as nonlinear descriptor system. Campbell [2] has also shown that the autonomous version of (1)

$$M\dot{x}(t) = A(x)+Bu(t) \tag{2}$$

often arises in the formulation of some nonlinear optimal control problems. System (2) can also appear as a reducing order model in singular perturbation techniques [2]. The linear version of (2) has received considerable attention in the literature [4], [5], [6], [7]. In what follows, we will be concerned with the system (2). The approach presented seeks to analyze (2) and to avoid utilizing most of the assumptions that have been reported by Campbell [2]. The main thrust of this paper is to use the matrix generalized inverse to transform (2) to an equivalent nonlinear system. The idea behind our procedure is to decompose the system into two subsystems of smaller order, one is an ordinary nonlinear dynamic subsystem and the other is an algebraic nonlinear subsystem. The paper is divided into four sections. In section II an introduction to the Drazin inverse is presented. In section III we use the theorem of the Drazin inverse as a basic tool to develop a reduced-order model. In section IV we introduce an example to support the usefulness of our procedure.

II. PRELIMINARIES

It is known [3] that for any square matrix $\tilde{M} \epsilon C^{nxn}$, there exists a nonsingular matrix P such that

$$\tilde{M} = P^{-1}\begin{bmatrix} D & 0 \\ 0 & N \end{bmatrix} P \tag{3}$$

where C is the set of complex numbers, $D \epsilon C^{rxr}$ is nonsingular matrix, and $N \epsilon C^{n-rxn-r}$ is nilpotent.

The smallest non-negative integer k such that $N^k=0$ is called index of $\tilde{M}$ for M singular. A nonsingular matrix can be regarded as having index 0.

If $\tilde{M} \epsilon C^{nxn}$ is given as in form (3), then the Drazin inverse of $\tilde{M}$, denoted by $\tilde{M}^D$, is defined as

$$\tilde{M}^D = P^{-1}\begin{bmatrix} D^{-1} & 0 \\ 0 & 0 \end{bmatrix} P \tag{4}$$

Note that the Drazin inverse is unique and satisfies

$$\tilde{M}\tilde{M}^D = \tilde{M}^D\tilde{M}, \quad \tilde{M}^D\tilde{M}\tilde{M}^D = \tilde{M}^D, \text{ and}$$

$$\tilde{M}^{\ell+1}\tilde{M}^D = \tilde{M}^{\ell} \text{ if } \ell \geq k.$$

If $\tilde{M}$ is nonsingualr, the N block is missing in (3) and $\tilde{M}^D = \tilde{M}^{-1}$, while if $\tilde{M}$ is nilpotent the D block is missing in (3) and $\tilde{M}^D=0$.

III. PROBLEM FORMULATION AND MODEL REDUCTION

Consider an autonomous nonlinear semistate equation given by

$$M\dot{x}(t) = A(x)+Bu(t)$$

where we assume that

(i) M is a singular square constant matrix .
(ii) x(t) and u(t) are vector valued functions of t and $B \epsilon C^{nxm}$ is a constant matrix.
(iii) A(x) is a nonlinear vector valued function of x that can be decomposed as

$$A(x) = A_o x+A_{oo}(x)$$

where $A_o\ C^{nxn}$ is a nonsingular constant matrix,

$A_{oo}(x)\ C^n$ is a nonlinear vector valued function of x(t).

Our viewpoint is that system (2) is composed of two subsystems whose semistate vectors are x, i.e., the nonlinear subsystem (2) is obtained as a result of incorporating a nonlinear dynamical subsystem with a nonlinear algebriac subsystem. Reflecting this viewpoint, we shall call the system given by (2) a composite system. The existence of assumption (iii) says that there is a nonsingualr matrix A_o such that the composite system can be rewritten as

$$\tilde{M}\dot{x}(t) = x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t) \tag{5}$$

where

$$\tilde{M} = A_o^{-1}M \tag{6-a}$$

$$\tilde{A}_{oo}(x) = A_o^{-1}A_{oo}(x) \tag{6-b}$$

$$\tilde{B} = A_o^{-1}B. \tag{6-c}$$

The following theorem will be essential in what follows.

Theorem: The general vector equation

$$\dot{x}(t) = \tilde{M}^D \left[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t)\right] + P^{-1}\begin{bmatrix} 0 & 0 \\ 0 & N^{k-1} \end{bmatrix} ph \tag{7}$$

satisfies (5) iff

$$(I - \tilde{M}\tilde{M}^D)\left[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t)\right] = 0 \tag{8}$$

for all t. Where I is the identity matrix, with appropriate dimension, $h \varepsilon C^n$ is an arbitrary vector, and $\tilde{M}^D$ is the Drazin inverse of $\tilde{M}$.

Proof: If (8) holds, then the vector equation

$$\dot{x}(t) = \tilde{M}^D \left[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t)\right] \tag{9}$$

satisfies (5). Conversely, if (9) satisfies (5), then

$$x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t) = \tilde{M}x(t) = \tilde{M}\tilde{M}^D \left[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t)\right]$$

$$\text{i.e., } (I - \tilde{M}\tilde{M}^D)\left[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u)\right] = 0$$

which is (8). Moreover it follows from

$$\tilde{M}P^{-1}\begin{bmatrix} 0 & 0 \\ 0 & N^{k-1} \end{bmatrix} ph = 0$$

and from (8) that every equation of the form (7) satisfies (5). On the other hand, let x(t) be any vector satisfies (5). Then for any arbitrary vector $d \varepsilon C^n$

$$\dot{x}(t) = \tilde{M}^D \left[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u\right] + P^{-1}\begin{bmatrix} 0 & 0 \\ 0 & N^{k-1} \end{bmatrix} pd$$

which is of the form (7).

In most important practical cases, index $(\tilde{M})=1$, therefore we can decompose the composite system into algebraic subsystems and nonlinear dynamical subsystem as follows:

Let $y(t) = Px(t) = [y_1(t), y_2(t)]^T$, $y_1(t) \varepsilon C^r$, $y_2(t) \varepsilon C^{n-r}$,

r is the rank of $\tilde{M}$, and "T" stands for transpose. Then (7) can be written as

$$\dot{y}(t) = P\tilde{M}^D \left[x(t) + \tilde{A}_{oo}(y_1,y_2) + \tilde{B}u(t)\right] + \begin{bmatrix} 0 & 0 \\ 0 & N^{k-1} \end{bmatrix} ph \tag{10}$$

which yields

$$\dot{y}_1(t) = [I\ 0]P\tilde{M}^D \left[P^{-1}(y_1,y_2)^T + \tilde{A}_{oo}(y_1,y_2) + \tilde{B}u(t)\right] \tag{11-a}$$

Since (8) must be satisfied for all t, then with index $(\tilde{M})=1$, we get

$$y_2(t) = -[0\ I]P\left[\tilde{A}_{oo}(y_1,y_2) + \tilde{B}u(t)\right] \tag{11-b}$$

where $\tilde{A}_{oo}(y_1,y_2)$ is a nonlinear vector valued function, P is a nonsingular matrix defined by (3). Equation (11) is simply a system of ordinary differential equations coupled to a system of algebraic nonlinear equations.

Assume (11-b) has at least one solution. Then substituting the solution of $y_2(t)$ in terms of y_1 and u from (11-b) into (11-a), we get a nonlinear differential equation solely in terms of $y_1(t)$. This equation is a classical nonlinear differential equation on which much has been written. Note that equation (8) absolutely defines a manifold (possibly nonlinear) of consistent initial condition. That is,

$$(I - \tilde{M}\tilde{M}^D)[x(o) + \tilde{A}_{oo}x(o) + \tilde{B}u(0)] = 0$$

In many cases M in equation (2) is given in a Jordan form with Index (M)=1. Then we could choose A_o such that $A_o^{-1}M=M$. In this case x(t) = y(t) and (11) can be rewritten as

$$\dot{x}_1(t) = [I\ 0]\ M^D[x(t) + \tilde{A}_{oo}(x) + \tilde{B}u(t)] \qquad (12\text{-a})$$

$$x_2(t) = -[0\ I][\tilde{A}_{oo}(x) + \tilde{B}u(t)] \qquad (12\text{-b})$$

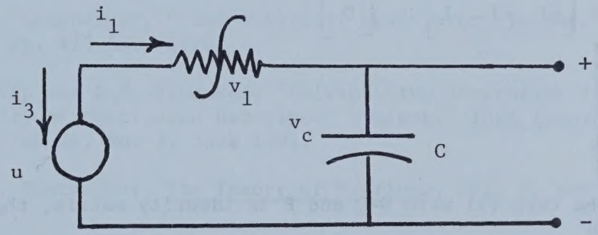
where

$$x_1 \varepsilon C^r, \quad x_2 \varepsilon C^{n-r} \quad .$$

As an illustration for the usefulness of the techniques presented in this section as an efficient tool in analyzing system (2), an example from [1] will be presented.

IV. EXAMPLE

Consider the following circuit



It has been shown in [1] that the above circuit admits no state equations with the nonlinearity:

$$v_1 = -3i_1 + i_1^3$$

The semistate equations for the circuit shown has been given by

$$\begin{bmatrix} C & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \dot{x} = \begin{bmatrix} 0 & 0 & -1 \\ -1 & -1 & 0 \\ 0 & 1 & -3 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ x_3^3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u \qquad (13)$$

where

$$x = \begin{bmatrix} v_c \\ v_1 \\ i_3 \end{bmatrix}$$

choose

$$A_o = A_o^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Then (13) can be rewritten as

$$\begin{bmatrix} C & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \dot{x} = x + \begin{bmatrix} -1 & 0 & -1 \\ 0 & 0 & -3 \\ -1 & -1 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ x_3^3 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

which is in the form (5), that is

$$\tilde{M} = M = \begin{bmatrix} C & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\tilde{A}_{oo}(x) = \begin{bmatrix} -1 & 0 & -1 \\ 0 & 0 & -3 \\ -1 & -1 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ x_3^3 \\ 0 \end{bmatrix}$$

$$\tilde{B} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Since $\tilde{M}=M$ is in the form (3) with N=0 and P is identity matrix, then one may use (12) to obtain

$$\dot{x}_1 = -C^{-1} x_3 \qquad (14)$$

$$\begin{bmatrix} x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -x_1 + u \\ \frac{1}{3} x_2 + \frac{1}{3} x_3^3 \end{bmatrix} \qquad (15)$$

Differentiating (15) gives

$$\dot{x}_2 = -\dot{x}_1 + \dot{u} = C^{-1} x_3 + \dot{u} \qquad (16)$$

and

$$\dot{x}_2 - 3\,\dot{x}_3 + 3\,x_3^2\,\dot{x}_3 = 0 \qquad (17)$$

Substituting (16) into (17) yields

$$(3\,x_3^2 - 3)\,\dot{x}_3 = -C^{-1}\,x_3 - \dot{u} \tag{18}$$

Equation (18) is a classical nonlinear equation which can be solved for most $x_3(o)$. For more details on the solution of (18) see [2].

It is worth noting to call the reader's attention that we obtain the same results for the solution of the above example as in [2]. One may use (12-b) or (15) to obtain the consistent initial conditions, so

$$\begin{aligned} x_2(0) &= u(0) - x_1(0) \\ x_2(0) &= 3\,x_3(0) - x_3^3\,(0) \end{aligned} \tag{19}$$

Thus for any arbitrary $\gamma = x_3(0)$, the initial condition given by (19) defines a nonlinear manifold described by

$$x(0) = [u(0) - 3\gamma + \gamma^3,\ 3\gamma - \gamma^3,\ \gamma]^T$$

V. REFERENCES

[1] R.W. Newcomb, "The Semistate Discription of Nonlinear Time-variable Circuits," IEEE Trans. Circuits Syst., Vol. CAS-28, No. 1, pp. 62-71, Jan. 1981.

[2] S.L. Campbell, "A Procedure for Analyzing a Class of Nonlinear Semistate Equations that Arise in Circuit and Control," IEEE Trans. Circuits Syst., Vol. CAS-28, No. 3, pp. 256-261, March 1981.

[3] S.L. Campbell, Singular Systems of Differential Equations, Pitman, 1980.

[4] D.G. Luenberger, "Time-Invariant Descriptor Systems," Automatica, Vol. 14, pp. 473-480, 1978.

[5] E. Yip and R.F. Sincovec, "Solvability, Controllability, and Observability of Continuous Descriptor Systems," IEEE Trans. Automat. Contr., Vol. AC-26, No. 3, June 1981.

[6] F.R. Gantmacker, The Theory of Matrices, Vol. 2, New York: Chelsea, 1974.

[7] G.C. Verghese, B.C. Lēvy and T. Kailath, "A Generalized State-Space for Singular Systems," IEEE Trans. Aut. Cont., Vol. AC-26, No. 4, Aug. 1981.

OPTIMAL SCHEDULING OF ASSEMBLY LANGUAGE KERNELS FOR VECTOR PROCESSORS

SIAMAK ARYA, D. A. CALAHAN
Department of Electrical & Computer Engineering
The University of Michigan
Ann Arbor, MI 48109

ABSTRACT

A mathematical programming model is developed to represent the optimal scheduling of addressing, scalar, and vector instructions on the CRAY-1. The model precisely accounts for all register, data path, functional unit, and memory protocols. Several example scientific kernels are scheduled.

I. OPTIMIZATION OF SCIENTIFIC KERNELS

As the size of scientific and engineering simulation problems increases, their computational complexity approaches an asymptotic behavior associated with increasingly lower level instruction loops. In the limit, a syntactically simple accumulation loop often dominates total computation time, especially in linear algebra and signal processing codes.

Unfortunately, the optimal control of data flow associated with loops that appear simple in a high level language may be nontrivial for memory-hierarchial functionally-concurrent processors, such as the Floating Point System 164 and the CRAY-1. In many important applications, only a fraction (1/5-1/3) of achievable performance is available from a high level language and one must resort to assembly language coding to preserve the efficiency of high performance algorithm development. Indeed, timing simulators have been developed for both of these processors, encouraging such programming by algorithm developers. Even with such aids, one is often left with the likelihood that the hand-scheduling of inner loop assembly instructions is suboptimal; often one could justify considerable effort in truly optimal scheduling of only 10-100 assembly instructions.

The instruction-scheduling literature has recently been concerned at a low level of control with microcode compaction [1], and at a high level with automatic vectorization [2] from a high level language. The scheduling procedures of this paper, although they are applied to vector and array processors, are at low level of control, and so are closest mathematically and functionally to the former. The major differences are the following:

1. The compaction goal of microprogram scheduling is replaced by a minimum-time criterion.
2. The mathematical modeling is extended to a much larger variety of register types and functional units associated with a vector processor, namely, the CRAY-1.
3. Instruction scheduling is characterized - apparently for the first time - as a mixed integer programming problem.

4. The optimal scheduling of instruction loops, important in scientific codes, is modeled in addition to the straight line code sequences associated with microcode.

II. CRAY-1 ARCHITECTURE AND CONFLICTS

The scheduling method to be introduced is a general one that can be applied to numerous pipelined architectures. For illustration, the CRAY-1 architecture and associated assembly language (CAL) will be used in the following discussion.

The CRAY-1 has the following architectural features.

1- 1 instruction issuer to all Functional Units (FU´s).
2- 5 types of register groups; Address (A), Scalar (S), Vector (V), B, and T.
3- 8 A, 8 S, 8 V, 64 B, and 64 T registers.
4- 13 pipelined FU´s, including the memory. Each pipeline stage is one clock long.
5- a set of access paths from FU´s to registers and vice versa.

The following general restrictions apply to the control of these resources.

1- At most one instruction can be issued at any clock period.
2- An instruction can be issued only if the registers and the FU required by the instruction have not been reserved by a previous instruction.
3- At most one register in A register group and one register in S register group can receive data at any clock period, since each one of these register groups has one update access path.
4- Vector instructions are the only instructions that can reserve the FUs, except for memory.
5- Result registers are reserved until the result is received.
6- Vector-operand registers are reserved for a known period of time, depending on the vector length.
7- No instructions are issued if a block transfer is in progress. i.e. issue is held if a set of B or T registers are being transfered to or from main memory.

There are other conditions which control instruction-issue that presently are not considered in the model, such as instruction buffer fetch in progress, exchange package in progress, and instruction not in the instruction buffer.

III. ELEMENTARY TIMING MODEL

The mathematical scheduling model is developed from data dependencies that infer, for a given CAL code, reservations on registers, FU´s and other resources. This establishes a graph and an elementary ´Minimum-Time´ scheduling problem. The model is then constrained by other architectural features to produce the complete mathematical programming problem. This section is concerned with the elementary model.

A. PAIRWISE DEPENDENCY

Definition 1- Given two instructions Ii and Ij, where Ij succeeds Ii in the sequential order of instructions, Ij is pairwise dependent on Ii iff at least one of the operand registers of Ij is the result register of Ii or the result register of Ij is one of the registers of Ii.

The pairwise dependency relation, also stated by [1] and [2] in a similar manner, implies that Ij should not be issued until Ii has been issued and the dependency condition has been satisfied. If the issue order is reversed, it will cause a change in the nature of the code. The pairwise dependency between two instructions is displayed by --p->; e.g. if I2 is pairwise dependent on I1, then I2--p->I1.

B. GLOBAL DEPENDENCY

Definition 2- Two instructions Ii and Ij are globally dependent if there exists Ik1, Ik2,..., Ikn such that Ij--p-> Ikn--p-> ...--p-> Ik2--p-> Ik1--p-> Ii .

If two instructions are globally dependent, this dependency enforces a specific issue-time-order on them to preserve the nature of the code. This relation is displayed by --g->; e.g. if I2 is globally dependent on I1, then I2--g->I1.

C. DEPENDENCY GRAPH

The dependency relation can be displayed by a directed graph. Each node of the graph represents an instruction, each edge specifies the dependency between the two nodes connected by that edge, the direction of each edge is from the predecessor to the successor, and the value on each edge represents the time that the successor must wait after the predecessor has been issued, before it can be isssued. Two nodes are called adjacent if they are connected by an edge. Two adjacent nodes are pairwise dependent. Two nodes are globally dependent if there exists a directed path from one to the other. Fig.1 shows an example of a set of instructions and its dependency graph. For example, in the sample instruction set of Fig.1a, I2--p->I1, I6--p->I4, and I6--p->I2; therefore, in its dependency graph of Fig.1b, nodes 2 & 1, 6 & 4, and 6 & 2 are adjacent nodes, and the numbers on these edges represent the length of their dependency in clocks. Also, since I6--p->I2--p->I1, then I6--g->I1.

D. MINIMUM TIME PROBLEM

Define the issue distance Dji as the minimum time between issues of Ij--p->Ii.

$$Tj - Ti \geq Dji \qquad (1)$$

where Tj and Ti are issue times of instructions j and i respectively. This formulation preserves the issue order of dependent instructions.

At this stage of the formulation, the formal minimization criterion is the time between the issue of the first instruction and the completion of the last instruction. That is termed the "minimum time" scheduling problem [3]. If D_{ji}'s are integers for all i and j, then the T_j and T_i are guaranteed to be integers as well.

```
0- ENTRY  =    *
1-        S1   /HS3
2-        S4   S2*FS1
3-        A1   A1+1
4-        S5   S1+FS1
5-        A3   A1+A2
6-        S3   S5+FS4
7-        ,A3  S5
```

a

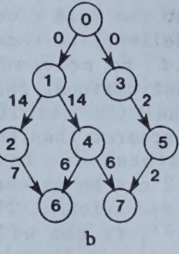


Fig.1- a. sample instruction set. b. dependency graph for a)

IV. SCHEDULING WITH CONSTRAINTS

In general, for a given set of instructions, not all of the instructions are pairwise or globally dependent on each other. Two instructions that are not dependent are called independent and can be issued in any order. For both dependent and independent instructions, it is necessary to comply with the restrictions imposed by the architecture. To include these restrictions in the model, a set of constraint equations are added to those represented by (1).

CHAINING CONSTRAINTS- Chaining on the CRAY-1 refers to the immediate transfer of data between pipelines involved in successive vector instructions. Such chaining requires that the second instruction be prepared to issue as the first result emerges (at chain slot time) from the first pipeline. For example, the following instructions have the potential to be chained.

```
V1    V2*FV3
V4    V1+FV0
```

Failure to chain requires that the first vector operation to be carried to completion; when the result vector is completely stored in a vector register, the second instruction can issue. Chaining is usually desireable, since both pipelines can be concurrently busied on different parts of a long vector.

Chaining is formulated as follows.

$$T_j - T_i \geq TR_i \quad \text{or} \quad T_j - T_i = TCHS_i \qquad (2)$$

where $TCHS_i$ is the chain slot time and TR_i is the time to complete the ith vector instruction, when chaining fails. Equation (2) eliminates

(1) for these instructions.

ACCESS PATH CONFLICT CONSTRAINTS- If both instructions are either A or S type, then it is possible that these instructions have access path conflict. This means that both instructions will finish execution at the same time and must deliver the result to two result registers of A or S group at the same clock. Since only one access path is available for result delivery to these register groups, one of the instructions must be held to prevent access path conflict. For two pairwise dependent instructions, this conflict is possible only when the number of clocks that the result register of the preceding instruction is reserved is greater than that of the succeeding instruction plus their issue time distance. In Fig.1a, if instruction 1 is issued at T1, instruction 2 can be issued at T1+1 or later; however, instruction 1 will finish execution at T1+13, and, if instruction 2 is scheduled to be issued at T1+7, it too will finish in 7 clocks at T1+13. Hence, to prevent the access path conflict for two pairwise dependent or two independent instructions, one of the following conditions should hold.

$$Ti + TRi - (Tj + TRj) \geq 1 \quad \text{or} \quad Tj + TRj - (Ti + TRi) \geq 1 \tag{3}$$

where TRi and TRj are the number of clocks that result registers of instructions i and j are reserved, respectively.

VECTOR-OPERAND CONFLICT CONSTRAINTS- If both instructions are independent and V type, vector-operand conflict can occur. According to definition 1, two independent instructions can share only their operand registers. For vector instructions , vector-operand registers are reserved and cannot be used by more than one instruction at any time. Therefore, the execution of these instructions cannot be overlapped if they share at least one vector-operand register. However, since the instructions are independent, either one can precede the other one. Hence, this type of conflict is formulated as follows.

$$Tj - Ti \geq TOPi \quad \text{or} \quad Ti - Tj \geq TOPj \tag{4}$$

where TOPi and TOPj are the number of clocks that the operand registers of instructions i and j are reserved.

FU CONFLICT CONSTRAINTS- For independent instructions, FU conflict can occur for either (a) two vector instructions using the same FU, (b) a vector instruction and a scalar instruction using the same FU, (c) a vector memory reference and any other memory reference, or (d) a block transfer and a memory reference. This conflict is similar to vector-operand conflict, in the sense that the execution of these instructions cannot be overlapped. Hence, the formulation is the same as (4) with TOPi and TOPj replaced with TFUi and TFUj, the number of clocks that the FU is reserved for instructions i and j, respectively.

ISSUE CONFLICT CONSTRAINTS- Since no more than one instruction can be issued at any clock period, it must be assured that independent instructions will avoid issue conflict. Most instructions require one or two clocks for issue. This conflict is identical to FU or vector-operand conflict, so it is formulated similarly by replacing TOPi and TOPj in (4) with TISSi and TISSj, the time to issue instructions i and j, respectively. For two independent instructions, if FU or vector-operand conflict occurs, issue conflict need not be considered. However, if access path conflict is possible, issue conflict cannot be ignored.

V. LOOP SCHEDULING

A set of instructions that are executed repeatedly and consecutively until a specific condition is satisfied is termed a loop. Execution of a loop is similar to execution of a code composed of a set of copies of the loop. The relative instruction issue times within the loop might be different between successive loop executions due to interloop dependencies.

Let it be assumed that a periodic steady-state loop execution schedule exists, with period Tx defined from

$$Tj - Ti = Tx \tag{5}$$

where Ti and Tj are issue times of the same (arbitrary) instruction in successive loops. The minimization of Tx is then the optimal scheduling criterion.

The location of the conditional loop JUMP(s) may be fixed or floating, as illustrated in Figure 2. The large brackets show the floating range of the loop-limits. When the JUMP is floating but is optimally scheduled in an unnatural position, loop pre- or post-processing code must be added to complete the loop logic.

```
0- ENTRY  =    *             0-          ENTRY  =      * -----,
1-        S1   /HS3          1-      S1  /HS3            .    |
2-        S3   S2*FS4        2-      S3  S2*FS4          .    |
3-        V1   ,A0,1         3-      V1  ,A0,1           .    |
4-        V2   V1*FV3        4-      V2  V1*FV3            -----+-,
5-        JAZ  ENTRY         5-              JAZ  ENTRY  -----'  |
                             6-      S1  /S3             .       |
                             7-      S3  S2*FS4          .       |
                             8-      V1  ,A0,1           .       |
                             9-      V2  V1*FV3            -------'

              a                            b
```

Fig.2 a. sample loop. b. 2 copies of a) with floating JUMP.

VI. INTEGER PROGRAMMING (IP)

Section IV. displays a set of inequalities that can be solved by an IP. The format of pairs of inequalities shown in (3) and (4), which are called " either or " inequalities, can be converted to an IP model as follows [3].

$$\begin{array}{l} T_i-T_j\geq a_j \\ \text{or} \\ T_j-T_i\geq a_i \end{array} \Longrightarrow \begin{array}{l} \theta*Y_{ij}+T_i-T_j\geq a_j \\ \\ \theta*(1-Y_{ij})+T_j-T_i\geq a_i \end{array} \Longrightarrow a_j\leq \theta*Y_{ij}+T_i-T_j\leq \theta-a_i \qquad (6)$$

where Y_{ij} can only take one of the values 0 or 1, and θ is a large positive integer. The effect of the Y_{ij} being 0 or 1 is to force a solution off of the hyperplane $T_i - T_j = 0$. Further, the T_i and T_j are assured to be integer if Y_{ij} equals 0 or 1. Thus, only the Y_{ij} need be forced to integers, yielding the name mixed integer programming.

The chaining inequalities of (2) also can be represented by such inequalities.

$$\begin{array}{l} T_j-T_i\geq a \\ \text{or} \\ T_j-T_i=b \end{array} \Longrightarrow \begin{array}{l} \theta*(1-Y_{ij})+T_j-T_i\geq b \\ \theta*(Y_{ij}-1)+T_j-T_i\leq b \\ \theta*Y_{ij}+T_j-T_i\geq a \end{array} \Longrightarrow \begin{array}{l} \theta*(1-Y_{ij})+T_j-T_i\geq b \\ a\leq\theta*Y_{ij}+T_j-T_i\leq\theta+b \end{array} \qquad (7)$$

VII. SCHEDULING EXPERIMENTS

The IP that has been used for scheduling of some sample programs is the IBM-MPSX/MIP/370. The package uses a two phase process to solve any given integer programming problem. In phase one, the problem is processed as a Linear Programming (LP) problem by MPSX/370. The second phase of the process uses the MIP/370, which is a mixed IP to generate integer solutions for the Y_{ij}. Each integer-feasible solution is displayed until the optimum solution is found. The package provides the options to limit the processing time and to replace the first phase by a user-provided starting point for MIP.

The results of three scheduling experiments are given in Fig. 3.

The first example is a scalar code with 12 instructions from a linear algebra code for LU factorization of a 2x2 matrix. The timing results for scheduled and unscheduled codes (Experiment 1 in Fig. 3) are not significantly different, since a high degree of dependency exists between instructions and rearrangement is limited. Experiment 2 involves the simultaneous factorization of 2x2 matrices. This finds application in the solution of (two) coupled partial differential equations, where different grid points are to be simultaneously eliminated but data storage or short vector length does not favor a vector mode solution. Two copies of the unscheduled code of Experiment 1 are concantenated, but the codes use different scalar registers and so are independent. The intention of scheduling is to interlace the independent instruction sequences. The results of Fig. 3 show that the timing is more than halved using the MIP scheduling.

Another important application in scientific codes is the scheduling of assembly-coded library routines. In Experiment 3, a 64-bit vector integer multiply, obtained from Cray Research, Inc., was rescheduled for a vector length of 64. The speedup was only 9%, since some care had already been taken with the hand-coded version. If this were representative of other achievable library speedups, however, the optimal scheduling would have to be considered well worthwhile.

Experiment #	# of CAL Instructions	Unscheduled Time (Cl)	Scheduled Time (Cl)
1	12	87	68
2	24	165	74
3	34	876*	798

Fig.3- Timing results for sample experiments in clocks. (* hand scheduled code).

VIII. CONCLUSIONS

A general software package is being developed to accept a CAL code as input data and then to generate the inequality information in a form acceptable to the MIP software; options include the ability to specify looped- or straight-line code optimization. Its major application will be the optimization of linear algebra codes.

Present limitations on the methodology include (a) dependence of the schedule on vector length, (b) an inability to reassign registers during scheduling, and (c) no means of trading off scheduling time and optimality of code. These issues will be addressed shortly.

ACKNOWLEDGEMENT

This research was supported by the Computational Mathematics Group of the Air Force Office of Scientific Research under grant AFOSR 80-0158.

REFERENCES

[1] Mario Tokoro, Eiji Tamura, and Takashi Takizuka, " Optimization of Microprograms ", IEEE Trans. on Comp., Vol. C-30, No. 7, July 1981
[2] David J. Kuck, " The Structure of Computers and Computations", Vol. 1, Wiley, N.Y., 1978
[3] Katta Murty, " Linear and Combinatorial Programing ", Wiley, N.Y., 1976

DESIGN OF A COMMUNICATIONS KERNEL FOR LOOSELY COUPLED MULTIPROCESSORS

Abhaya Asthana
Charles S. Roberts
G. Kenneth Swanson

Bell Laboratories
Holmdel, New Jersey 07733

ABSTRACT

Although the inherent data transfer rates offered by the physical communication medium in local networks can be 1-300 Mbits/sec, the actual data transfer rates perceived by the end processes are orders of magnitude lower. Substantial progress has been made in designing networks to interconnect processors, but the problem of finding efficient primitives for reliable interprocess communication is far from solved.

In this paper we present the design of a communications kernel that implements a set of efficient primitives for reliable interprocess communication. The communications kernel serves as the lowest layer in a distributed operating system research facility being designed and implemented for loosely coupled multiprocessors, in our case, a heterogeneous mix of computers (VAX-11, PDP-11 and Motorola MC68000) interconnected via a high speed bus.

1. INTRODUCTION

The advent of relatively inexpensive, high bandwidth local networking hardware has made it possible to experiment with operating system architectures that allow application programs to be subdivided into modules which execute concurrently on multiple processors. A common way of organizing such distributed operating systems is as a collection of *processes* that communicate with each other primarily via *messages* [2,5,9,11,12,16,18,23]. In such schemes the mechanisms for interprocess communication (i.e. message passing) often assume a preeminent role in determining the functionality, performance and, ultimately, the overall success of the operating system.

The raw bandwidth of the physical communication medium used in local networks is typically in the range of 1-300 MBits/s. However, the effective data transfer rates that end processes perceive are several orders of magnitude lower. It is now widely recognized that, in many cases, the cause of these slow transfer rates is not the physical communication medium but the processing overhead for protocols required at the ends to transform the physical medium into a reliable and useful communication channel. A major barrier to efficient and effective distributed computing is this gap between the raw transfer rate and the effective transfer rate.

By properly partitioning the operating system code, it should be possible to move the communications function into a separate processor and to build hardware to improve its performance. To achive this goal our operating system code is partitioned into two sections: the *Executive* and the *Communications Kernel.* This paper describes a set of interprocess communication primitives. In addition, we describe the design of the communications kernel — a set of modules that efficiently and reliably implements the aforesaid interprocess communication primitives.

The work described in this paper is part of a larger research project [10] whose goal is to design and implement a more complete operating system experiment for a collection of loosely coupled multiprocessors. The approach in this paper, however, has been to regard the communications primitives as being the most fundamental and to first organize and define the concepts at this level before proceeding on to other operating system issues. The goal of the authors and their colleagues has been to gain an understanding of how to build distributed operating systems and to exploit hardware to produce a system organization that is modular, effective and robust.

1.1 Hardware Organization

The organization of the underlying hardware, whose physical structure ressembles that of a network, is shown in Fig. 1. The system consists of a collection of up to 256 autonomous machines interconnected by a high speed bus. The machines connect to the bus via a Bus Interface Unit (BIU), and the link that connects a machine to a BIU may be serial or parallel and may range from a few meters to several hundred meters in length. The basic unit of data transfer between machines is a *packet.* A Store Group arbitration scheme [1] is used for allocating the bus bandwidth between machines. This bus arbitration feature is one of the things that distinguishes this hardware scheme from other networks such as Ethernet [14].

Each machine itself is a coprocessing system consisting of two processors — a Communication Engine (CE) that is responsible for all communications activities in that machine, and a Processing Engine (PE) that is responsible for managing the application processes. We will not concern ourselves in this paper with how the operating system modules are subdivided between the CE and the PE. Instead, we will treat the CE and PE together as a single logical entity.

1.2 Operating System Structure

The operating system is partitioned into two layers: the *communications kernel* (CK) and the *executive* (EX). The objective is to create an environment in which modules may communicate without concern for the topology of the system as a whole, using as few mechanisms as possible.

The EX provides the environment in which user processes are executed. It performs functions such as creation and termination of user processes and establishment and takedown of interprocess communication channels.

The CK allocates the machine resources consisting of memory, cpu, devices and communications channels. Process dispatching and interrupts are also handled by the communications kernel. Its major function, however, is to provide reliable interprocess communication (IPC) primitives.

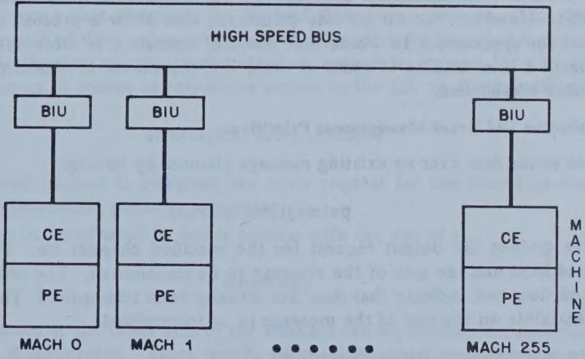


FIGURE 1 HARDWARE ORGANIZATION. BIU = BU INTERFACE UNIT, CE = COMMUNICATION ENGINE, PE = PROCESSING ENGINE.

1.3 Message Channels

The communications kernel supports only one mechanism for all interprocess communication: *message channels.* The properties of message channels are similar to pipes in the UNIX* operating system [19] with several distinctions. First, framing information (i.e. the message boundaries) is preserved in a message channel; in contrast, communication over pipes is in the form of unframed byte streams. To obtain an unframed byte stream from a message channel, the receiving process may simply ignore the framing information. Second, message channel may be established between any two processes that wish to communicate. There is no common parentage restriction as in the case of pipes. Third, message channel provides a path for full duplex communication, which in the UNIX operating system can be achieved only by setting up one pipe for each direction of data transfer. In principle, a message channel is similar to the harware concept of a *virtual circuit.* Both provide a connection type of service. However, we prefer the term message channel to describe our logical construct because it is more descriptive of the underlying communication mechanism.

In our communication model the communication path is created by mutual consent of the two entities desiring to communicate. Once the channel is established, the end processes can interactively engage in rapid and reliable exchange of messages. When the channel is no longer needed it may be removed by either process. The period between setup and takedown of a message channel is a *session.* In this respect a message channel is analogous to a telephone system style of communication, while systems based upon datagrams, on the other hand, conform more to to a post-office model.

In this paper we are primarily concerned with the design of the communications kernel. Section 2 presents a description of the interprocess communication primitives. In section 3 we describe the implementation of these primitives in moderate detail. The last section provides a discussion of our reasons for selecting certain choices.

2. INTERPROCESS COMMUNICATION PRIMITIVES

Before we proceed with a discussion of the IPC primitives, we briefly define the primary objects with which the CK deals: *processes, message channels, messages,* and *events.* A *process* is an executing image of a program. A *message channel* is a connection between two processes over which they can interact. A *message* is the unit of information exchanged between processes over a message channel. Finally, *events* are activities on which processes synchronize: for instance, completion of a communication request or the arrival of a request for a message channel setup.

The interprocess communication provided by our communications kernel is asynchronous (non-blocking). However, we do provide primitives that allow a process to achieve synchronous behavior for applications in which that mode of operation is more suitable. In what follows, we present a brief outline of some of these IPC primitives. It is assumed that the session has already been established.

2.1 Communication and Event Management Primitives

A process sends data over an existing message channel by issuing:

putmsg(lcn, iostruc)

This primitive queues the output request for the specified channel *lcn.* The structure *iostruc* specifies the address and the size of the message to be transmitted. The return from putmsg is immediate and does not indicate that data has actually been transmitted. The primitive putmsg places no constraints on the size of the message to be transmitted.

To receive data a process issues the request:

getmsg(lcn, iostruc)

---

* UNIX is a trademark of Bell Laboratories.

The address and size of the buffer in which the message is to be received are specified in *iostruc*. Once again the primitive returns immediately and does not indicate that data has been received. The determination of whether an input/output operation has completed is made by testing a flag in the input/output structure *iostruc*. A flag in the status structure indicates whether the end of message has been reached or not; therefore, in case the receive buffer is too small to accommodate the message to be delivered, the process simply places another read to get the remaining bytes.

The primitives described so far are asynchronous; hence, they do not block the calling process. If the process wishes to synchronize on an input/output event, the waitio primitive is used:

waitio(lcna)

The calling process is blocked until a data transfer initiated by a previous getmsg or putmsg completes on any of the channels specified by the array *lcna*.

Another primitive

setvec(lcn, func)

allows a process to set up a software interrupt handler for a specified channel. Completion of an input/output operation on that channel causes the specified procedure *func* to be invoked.

Closely related to interprocess communication is the problem of addressing and naming. A name is simply a symbol (e.g. a text string) that identifies an object such as a process or a device. An address, on the other hand, is a number that specifies the physical location of an object. The mapping of names into addresses is a higher level function since the syntax and semantics of names depend entirely on what types of entities are being named and what use is being made of them. Therefore, a more elaborate discussion of this topic is beyond the scope of this paper. Suffice it to say that processes communicate over message channels, each end of which is identified by a *channel number*. The channel numbers on each end are normally not identical.

2.2 Session Creation and Termination

The request to establish a message channel to another process is made by:

lcn = open(dname)

where, *dname* is a name that identifies the desired destination process. This request is forwarded to the the specified destination process over a predefined channel that every process has to its EX. The destination process may accept or refuse to engage in a session with the specified source process. It makes its intentions known to the EX via the primitive:

answer(lcn, accept/refuse)

If the incoming setup request is accepted, the open request for the source process completes and the channel establishment phase is complete.

The session may be terminated by either process with the use of a

close(lcn)

primitive. The process at the other end of the channel can be notified via an interrupt that the channel has been disconnected. More details of message channel creation are contained in [10].

3. IMPLEMENTATION

From a communications system perspective the CK is organized in a protocol hierarchy as shown in Fig. 2. There are architectural similarities between the model shown in Fig. 2 and the ISO model [24]. However, there is no correspondence between the implementation complexities represented by the levels shown in our model and that of a typical long-haul network. The lowest level is the packet transport mechanism that, in our case, is a high speed bus. The protocol at this level specifies the hardware interfaces, timing characteristics, bit encoding, and line control procedures. The link level provides for the transmission and correct reception of packets to and from the bus. The packet protocol ensures orderly and reliable transmission and reception of packets over message channels. The interprocess communication primitives provide for movement of messages between processes. Finally, there may be a variety of protocols built upon the communication primitives that establish the procedures for how processes interact, for example, the conventions for file related services. In the following sections we outline the implementation of the link and packet level protocols.

3.1 Link level

The interface between the packet level and the physical transport is a set of drivers. A driver receives a packet and delivers it to the bus. A checksum is appended to the end of every outgoing packet. The driver signals the bus via control lines indicating that all the bytes in a packet have been written. On the receiving end the driver is interrupted, the packet read in, checked for errors and, if valid, passed on to the next level. Damaged packets are discarded by the driver. No retransmission is done at this level; the assumption being that the error rate of the underlying medium is sufficiently low.

3.2 The Packet Level

We use a combination of state transition diagrams and program variables [3] to describe the implementation of the packet protocol.

The packet format is shown in Fig. 3, and consists of the following fields:

- — Source and the destination addresses.
- — A control field that contains the packet type and other control bits.
- — A field specifying the number of data bytes within a packet.
- — The data field containing the actual data being transported.
- — A checksum code generated over the packet header and the data fields.

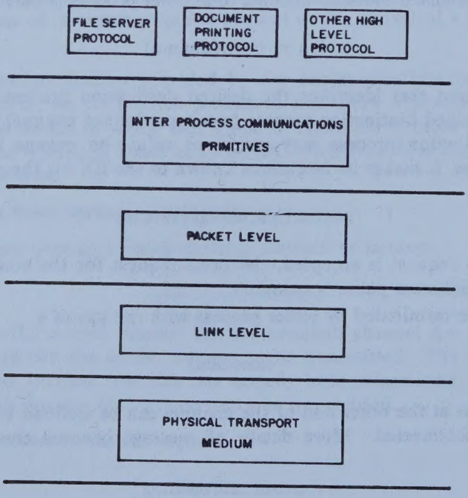

FILE SERVER PROTOCOL
DOCUMENT PRINTING PROTOCOL
OTHER HIGH LEVEL PROTOCOL
INTER PROCESS COMMUNICATIONS PRIMITIVES
PACKET LEVEL
LINK LEVEL
PHYSICAL TRANSPORT MEDIUM

FIGURE 2 COMMUNICATION PROTOCOL HIERARCHY

All the packets are fixed in format and size. The bits in the packet control field are interpreted as follows:

TYPE	This field specifies the packet type. In this paper we will consider the packet types: DATA, PROBE, and REJECT. DATA packets are the only ones that carry actual user data. The PROBE packets are used by the CK to poll the other end or to respond to a poll. The PROBE packets are necessary only when that information cannot be piggy-backed on a DATA packet. A REJECT packet is a special control packet used to request retransmission of lost packets.
PS	The *send sequence* number uniquely identifies to the receiver a transmitted data packet on a channel. With a PROBE or a REJECT packet, the send sequence number specifies the number of the next data packet to be transmitted.
PR	The *receive sequence* number specifies to the sender the next packet number that the receiver expects to receive. This implicitly acknowledges that all packets with sequence numbers less than PR have been received successfully.
FP/LP	These two bits indicate whether the packet is the *first* packet of a message, the *last* packet of a message or both the first and the last.
CF	This bit, the *command flag* bit, is set in a packet to extract a response from the receiver.
RF	The *response flag* bit is set in an outgoing packet *only* in response to a packet received with the command bit set.
SG	The stop/go bit is used for flow control on a message channel. The bit is normally zero (GO). If the receiver runs out of resources to receive incoming data it notifies the sender by setting the bit (STOP). The bit remains set in every outgoing packet as long as the receiver is not ready. A transition back to zero will result in resumption of traffic from the sender.

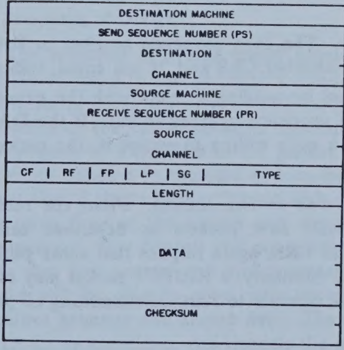

DESTINATION MACHINE
SEND SEQUENCE NUMBER (PS)
DESTINATION
CHANNEL
SOURCE MACHINE
RECEIVE SEQUENCE NUMBER (PR)
SOURCE
CHANNEL
CF | RF | FP | LP | SG |
TYPE
LENGTH
DATA
CHECKSUM

FIGURE 3 PACKET FORMAT

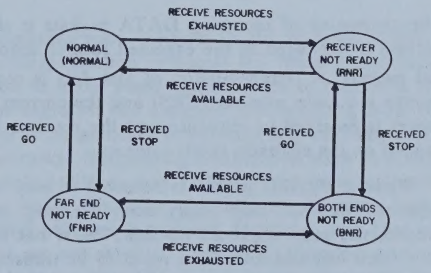

RECEIVE RESOURCES EXHAUSTED
NORMAL (NORMAL)
RECEIVER NOT READY (RNR)
RECEIVE RESOURCES AVAILABLE
RECEIVED GO
RECEIVED STOP
RECEIVED GO
RECEIVED STOP
RECEIVE RESOURCES AVAILABLE
FAR END NOT READY (FNR)
BOTH ENDS NOT READY (BNR)
RECEIVE RESOURCES EXHAUSTED

FIGURE 4 STATE TRANSITION DIAGRAM

Program Variables

In virtual circuit mode both ends of the circuit maintain some state information for the duration of the session. The following program variables are associated with each end of a virtual circuit.

CSS	*Channel Send Sequence Number:* This variable contains the sequence number of the next packet to be transmitted. The number is reset to zero at initialization. It is incremented by one whenever a packet is transmitted and is adjusted in case of retransmissions.
CRS	*Channel Receive Sequence Number:* This variable holds the sequence number of the packet that the receiver expects to receive. This variable is incremented whenever a packet is received in sequence and accepted.
CPR	*Channel Previous Receive Sequence Number:* This variable is used to manage packet acknowledgements. The receive sequence number contained within a packet implies to the sender that the receiver has correctly received all packets with sequence numbers less than that value. The CPR is set equal to the receive sequence number in the incoming packet. At any time the difference between CPR and the received sequence number PR is the number of packets being acknowledged, and the difference between CSS and CPR is the number of packets transmitted but not acknowledged.
STATE	*Channel State:* This variable holds the channel state. For active channels there are four possible states as shown in Fig. 4. No DATA packet is accepted if the state is either RNR or BNR. A transition to NORMAL or FNR is made when the receiver has acquired resources in which to receive incoming data. The sender is notified of this change in state via the SG field in the packet header. This notification takes place at no extra cost if there is data waiting to be transported to the other end; otherwise, an extra PROBE packet is required.

In addition to these variables there are several pointers maintained for incoming and outgoing messages. For example, an outgoing message has two pointers associated with it: one that points to the data that has to be transmitted and the other to the data that has been transmitted but not acknowledged. Finally, a single command timer is used to recover from failures.

Input Processing

The processing of incoming DATA packets is simple. The send sequence number in DATA packets is compared to the expected receive sequence number CRS and, if not equal, indicates lost packets. A retransmission of lost data is requested by sending a reject with the expected receive sequence number (CRS) and the current send sequence number (CSS). If the DATA packet is received in sequence and the receiver is in a state willing to accept it, the packet is queued on the channels receive queue.

When a PROBE packet is received it may cause one or two actions. First, the receive sequence number may imply acknowledging previously sent packets as described earlier. Second, the send sequence number, PS, if not equal to CRS, again implies that some packets have been lost and calls for a reject to be transmitted. Similarly, a REJECT packet may cause both of the above actions and, in addition, instructs the receiver to start retransmitting all packets starting with the send sequence number (PR).

Output Processing

Whether or not a packet can be sent is determined by the state of the receiving end and the number of unacknowledged packets. How often the remote end is polled for an acknowledgment influences the delay and the throughput perceived by the originator of the message. Two cases are possible depending on the value of the parameter Mx, which is defined as the the upper bound on the number of unacknowledged packets. The implementation allows either acknowledgment scheme to be used by setting the parameter Mx accordingly.

Case A. Mx = 1

In this case an explicit acknowledgment is demanded every packet by setting the command flag in every data packet transmitted. The receiver will be forced to respond with an explicit PROBE packet if no traffic is present in the reverse direction.

Case B. Mx > 1

In this case an acknowledgment is demanded only if the number of unacknowledged packets is equal to Mx or if the packet being sent is the last packet in a message. This represents a reasonable compromise. The size of the sequence number field determines the upper bound on the value of Mx. By making Mx large the number of acknowledgments required per message can be reduced. The drawback, however, is in the amount of retransmission required in case of packets lost due to errors or buffer unavailability.

The protocol allows for a certain amount of read ahead at the receiving end. This is desirable, since the sender is not inhibited from transmitting even though the receiver may not have issued a read. The buffer availability constraints do not allow for an unlimited read ahead. The length of the receive queue is what control the flow of data between the sender and the receiver. Two cases result depending on the value of a parameter Mr, the upper bound on the length of the read ahead queue.

Case A. Mr = 0

This is the degenerate case with no read ahead where the reception is completely receiver controlled. The sender cannot transmit unless the receiving end process allocates a receive buffer.

Case B. Mr > 0

In this case the sender can transmit a second message immediately after the first one. Assuming that a new read is not instantaneously issued, the packets will be saved on the receive queue until the read is placed. In this scheme it is possible for the read to complete immediately if the waiting message was short enough to be accommodated in the read ahead queue.

4. DISCUSSION AND CONCLUSION

The design and development of any large software system is an exercise in iterative design: issues are identified, alternative solutions explored, and trade offs are made among different criteria. In this section we present a discussion of some of the issues and point out our reasons for making certain choices.

4.1 Simplicity of Design

The presence of a high speed physical communication medium in the network makes it possible to simplify the underlying low-level communication protocols [6]. Since transmission bandwidth is plentiful, there is little motivation to make small optimizations to maximize the utilization of the communication medium. For example, the marginal gain resulting from the conservation of a few bits in the packet header is insignificant in our context, whereas, it is a significant concern in the case of long-haul networks. Simplifying assumptions such as a fixed packet size, a single header format with fixed field positions may, in fact, result in reducing the processing time at the source and destination ends. Similarly, the need for complex buffer management, flow control and congestion control schemes is reduced. However, this does not mean that flow control and buffer management are not required at all. The transport protocol should still be able to cope with mismatch between the rate at which two communicating machines generate and absorb data. The problem of buffer availability, both at the sending and the receiving machines, still impacts the protocol design.

4.2 Datagrams vs. Message Channels

We selected message channels as the fundamental and the only mode of communication between user processes. An alternative form of communication is datagrams. Datagrams travel one way and are not acknowledged, therefore, are simple to implement because system is not required to maintain state information. In datagrams, however, the issues of reliability are left to the end processes. Message channels, on the other hand, are designed to be reliable and to deliver data in the order sent. Implementation of message channels is slightly more complicated than datagrams since state information has to be maintained.

4.3 Flow Control

Flow control is the mechanism to regulate the behavior of a specific source and destination pair so that the source does not send data at a rate greater than the receiver can process it. An objective of flow control is to maintain traffic within limits compatible with the amount of available resources. This objective could be achieved simply by putting drastic limits on the sender's rate, but throughput degradations would be significant.

There are two basic mechanisms to regulate flow: credits and stop/go [8,17]. In the credits mechanism the sender is not allowed to send unless he has received from the receiver an indication of the amount of traffic that it can accept. This mechanism is useful if the transport delay is significant. Since in our network the transport delay is not significant, the communications kernel employs a simple stop/go scheme for flow control. In this scheme the receiver indicates to the sender via a signal when to stop sending traffic and when to resume transmission again.

If the signalling messages are not protected, the flow control schemes may fail. To safeguard against this the CK includes the stop/go bit in the control field of every packet.

4.4 Recovery/Reliability

There are two questions with respect to reliability: how much reliability should be provided and where should it be provided? In our design the responsibility of providing robust communication resides with the CK and is not left to the end processes. This is in contrast to certain systems based on datagrams [4, 22] where the issues of reliability are left to the end processes. The price of providing absolutely reliable system may be increased delay. On the other hand, reliability to a large degree is required in our system [10] because connections such as those between a process an a file server or between a process and a terminal depend on a reliable path.

4.5 Previous Work

One of the earliest experiments in this area was the Distributed Computer System of Farber [7], in which he proposed a ring structure for interconnecting processors. Wilkes and Needham [22] have followed a similar philosophy. Ethernet [14], on the other hand, represents a bus structure and uses a contention protocol as a low level communication mechanism. For certain classes of applications the Ethernet may represent an attractive solution as evidenced by Shoch [20]. However, for other classes of applications, characterized by a large number of cooperating processes with very close interaction, Ethernet may not be the most desirable communication medium. Other efforts in this area have been described in [13,15,21].

The presence of such hardware interconnects and the availability of inexpensive processing units also spurred a lot of interest in the development of operating systems. The motivation behind these systems is to provide a coherent environment to couple the loosely connected hardware units into a logical whole. Some noteworthy efforts in this direction have been DEMOS [2], Medusa[16], PILOT [18], StarOS [9], MININET [12], Hydra [23], and THOTH [5].

4.6 Conclusion

We have presented in this paper the design of a communications kernel that implements a set of effective primitives for reliable interprocess communication. In our work, as in this paper, we have taken a different, albeit simple, approach to designing distributed systems. We start with the premise that in a distributed environment effective and robust communication is more fundamental than other facilities that an operating system typically provides. Only when we have sufficiently understood the communications aspects of such a system, do we delve into the more ambitious experiments of "distributed computing".

REFERENCES

[1] S. R. Ahuja, "A group allocation scheme for high speed digital busses," Internal Memorandum, Feb. 1981.

[2] F. Baskett, J. H. Howard, and J. T. Montague, "Task communication in DEMOS," Proc. 6th Symposium on Operating System Principles, ACM, (Nov 1977), pp. 23-32.

[3] G. V. Bochmann and C. Sunshine, "Formal methods in communication protocol design," IEEE Trans. Comm., 28, 4 (April 1980), pp. 624-631.

[4] D. R. Boggs et. al., "Pup: An inter network architecture," IEEE Trans. Comm. 28, 4 (April 1980), pp. 612-624.

[5] P. R. Cheriton, et. al., "THOTH, a portable real time operating system," CACM 22, 2 (Feb 1979), pp. 105-114.

[6] D. Clark, et al., "An introduction to local area networks," Proc. IEEE, 66, 11 (Nov 1978), pp. 1497-1517.

[7] D.J. Farber, et al., "The Distributed Computing System," Proc. COMPCON, IEEE, March, 1973, pp. 31-34.

[8] M. Gerla and L. Kleinrock, "Flow Control: A comparative survey," IEEE Trans. Comm. 28, 4 (April 1980), pp. 553-574.

[9] A. K. Jones et. al., "StarOS: multiprocessor operating system for the support of task forces," Proc. 7th symposium on operating systems principles, ACM, (Dec 1979), pp. 117-127.

[10] H. P. Katseff, T. B. London and C. S. Roberts, "Gaggle - A Multi-Processor Operating System Based on Communications Channels," Internal Memorandum, April 1981.

[11] B. Liskov, "Primitives for Distributed Computing," Proc. 7th symposium on operating system principles, ACM (Dec 1979), pp. 33-42.

[12] J. Livesey, "Interprocess communication and Naming in the MININET system," Proc. Spring COMPCON, IEEE, (Feb 1979), pp. 222-229.

[13] E. G. Manning and R. W. Peebles, "A homogeneous Network for Data Sharing Communications," Computer Networks, 1, 4 (May 1977), pp. 211-224.

[14] R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," CACM 19, 7 (July 1976), pp. 395-403.

[15] D. L. Mills, "An overview of the Distributed Computer Network," Proc. NCC, 45, AFIPS, (June 1976), pp. 523-531.

[16] J. K. Ousterhout, et. al., "Medusa: an experiment in distributed operating system design," CACM, 23, 2 (Feb 1980), pp. 92-105.

[17] L. Pouzin and H. Zimmermann, "A tutorial on Protocols," Proc. IEEE. 66, 11 (Nov 1978), pp. 1346-1370.

[18] D. D. Redell et. al., "Pilot: An operating system for a personal computer," CACM, 23, 2 (Feb 1980), pp. 81-91.

[19] D. M. Ritchie and K. Thompson, "The UNIX Time Sharing System," CACM 17, 7 (July 1974), pp. 365-375.

[20] J. R. Shoch and Jon A Hupp, "Measured performance of an Ethernet local network," CACM, 23, 12 (Dec 1980), pp. 711- 721.

[21] R. J. Swan, S. H. Fuller and D. P. Siewiorek, "Cm* - a modular multiprocessor," Proc. AFIPS Press, Arlington, Va. 1977, pp. 637-644.

[22] M. V. Wilkes and R. M. Needham, "The Cambridge model distributed system," Operating Systems Review, 14, 1 (Jan 1980), pp. 21-29.

[23] W. Wulf, et. al., "Hydra: The kernel of a multiprocessor operating system," CACM, 17, 6 (June 1974), pp. 337-345.

[24] H. Zimmermann, "OSI Reference Model - The ISO Model of architecture for open systems interconnection," IEEE Trans. Comm. 28, 4 (April 1980), pp. 425-432.

APPROXIMATE ANALYTICAL PERFORMANCE EVALUATION IN BUS ORIENTED MULTICOMPUTER SYSTEMS

S. VIJAY IYENGAR and L. L. KINNEY
Department of Electrical Engineering
University of Minnesota
Minneapolis, MN 55455

ABSTRACT

This paper presents an analysis method for Markov models of multiple bus multicomputer systems. Markov models of such systems tend to have a large number of states. We present an approximation technique to reduce the number of states required in the analysis. Two types of systems are used as examples to illustrate the method. Both systems are loosely coupled systems with messages exchanged through common memory in the first system and passed from processor to processor in the second. A uniform reference model is used to describe the flow of interprocessor messages. Results of simulation are also given to show the accuracy of the approximations made in the analysis.

I. INTRODUCTION

Communication in multicomputer systems often uses a system of busses [1,2]. Considerable statistical analysis has been done for tightly-coupled systems using crosspoint switches to connect processors and memory [3-5]. More recently single bus systems have been analyzed [6,7]. A Markov model analysis of a homogeneous bus system consisting of more than one bus may lead to a large number of states in the model. We present an approximate model by reducing the number of states and approximating state transition probabilities. Simulation of the original system is done to indicate the accuracy of these approximations. The two systems used as examples in this paper do not necessarily represent any particular real-life systems but are used merely as vehicles to indicate the analysis techniques.

II. THE SYSTEMS

Block diagrams of the two systems are shown in Figures 1 and 2. In both examples each processor has its own clock and the synchronization of communication is handled by the bus interface units. In both examples the processors execute programs stored in private memories. In System I, messages are only transferred directly between the processors and the common memories, which are used as mailboxes. In System II, a processor sends messages to the bus interface unit of the other processors. The bus interface unit then transfers the message to the private memory of the receiving processor. Also, the references to private memory from the bus interface unit and from the local processor are totally transparent to each other. This kind of interleaving is possible with microprocessors like the 8080 and 6800, using fast memories and some buffering capability in the Bus Interface Unit.

The analysis in this paper is done assuming that all processors, memories and buses are identical. This is implicit in the assumptions of uniform request and service statistics. The method, however, can easily be extended to heterogeneous systems. The number of states needed for the analyses would, however, increase.

In both examples, the bus arbitration procedures used are intuitively efficient. A central queue is maintained of all unfulfilled requests. In System I, whenever a bus is to be allotted, the queue is scanned for the first processor that needs a common memory module that is free at that instant of time. In System II, we will call the computer (processor, memory and bus interface unit) requesting the bus, the source, and the bus interface unit being requested, along with its local processor and memory, the destination. By our previous discussion of transparency, a computer can be a source and a destination at the same time. A free bus, in System II, is allocated to the first processor in the queue that is requesting a free destination.

III. THE MODELING ASSUMPTIONS AND APPROXIMATIONS

It is difficult to include all the intricacies of an actual system. Some approximations are required to make the analysis tenable. These assumptions and approximations are discussed next. The models used for Systems I and II will be called Models I and II, respectively.

Assumption 1) Reference Model - A uniform reference model is used. Hence, a processor will request any of the common memory modules with equal probability in Model I, and in Model II a processor will request any of the other processors with equal probability. Unlike tightly coupled systems where locality of references is relevant, here it is not and the uniform reference model is reasonable.

Assumption 2) Request Statistics - The memoryless exponential distribution with mean rate λ is chosen for the request time, i.e., the time after a service until the next request. When the processor is queued due to a request not granted, a processor does not perform any useful computation.

Assumption 3) Service Statistics - The service time is the time spent transmitting messages. The effective service time distribution is assumed to be exponential with the mean time $1/\mu$.

Assumption 4) The arbitration time is assumed to be instantaneous. This approximation is reasonable if the time to arbitrate is much smaller than the mean request and service times.

Approximation Approximations in State Transitions - To analyze queueing systems using a first order Markov chain we need to define the states such that the probabilities of transition are determined by the current state. This generally requires a very large number of states. A state in each of our models is the doublet (I,J) where I is the number of buses in use and J is the number of processors queued. This definition does not permit a one-step Markov model because the transition probabilities depend on past history in some cases. The details of the one-step approximation to the actual systems are illustrated by considering one example in each model.

Consider an example with 5 processors, 4 memory modules and 3 buses in Model I (a 5 × 4 × 3 system). If we are in state (2,2), the next transition probability is approximated by assuming that the two queued processors can only request the two modules currently in use and by assigning equal probabilities to each possibility. This will not be correct if state (2,2) was the successor of the state (1,2) after an allotment of a bus to a new requesting processor. Since (1,2) has to be valid state, the 2 queued processors must be requesting the memory module in use and they will continue requesting that memory module even after the transition to state (2,2). Hence, assigning equal probabilities to all valid possibilities of memory references of the queued processors is a sort of averaging assumption. States with all the buses active, e.g., (3,1) for example, can be analyzed in two ways that we call continuous and discontinuous. In continuous analysis, we assume that each of the queued processors is requesting access to one of the active memory modules. In the example considered this is incorrect if the state (3,1) was entered from the state (3,0). The new queued processor could have requested any of the four memory modules in the system and yet the system will enter the state (3,1). In discontinuous analysis we assume that the queued processors could be requesting any of the memory modules in the system. In the example considered, this is incorrect if the state (3,1) was entered from the state (2,1). The queued processor could only be requesting one of the two active memory modules in the state (2,1) and will continue to do so after the system enters the state (3,1). These two approximations represent two extremes of the actual system.

The approximation in Model II is similar. However, if one of the queued processors is a destination currently in use, then it can only be requesting the other destinations being used. Again the actual destinations of the processors would depend on the past history of the system. The states with all the buses active can be handled in two ways, again, continuous and discontinuous. In continuous analysis, the references are limited to the set of active destinations and in discontinuous the references are allowed to range over all possible destinations in the system.

These approximations, as mentioned earlier, are made to reduce the number of states to a manageable number and are justified by obtaining results that are within a reasonable error of results for a simulation of the models without the approximation.

IV. ANALYSIS - MODEL I

Let 'P' be the number of processors, 'M' the number of memory modules and 'B' the number of buses, in the system. In general, the number of buses would be at most equal to the minimum of the number of processors and the number of memory modules.

As discussed earlier, our state is defined by the doublet (I,J), where

$$0 \leq I \leq B \tag{4.1}$$

$$0 \leq J \leq P-I \tag{4.2}$$

When I = 0, J can only be zero as any other value would mean a queue existing when none of the buses are being used, which is impossible, because arbitration is instantaneous. The number of states 'NT' for a P × M × B system is given by the following expression.

$$NT = [(P+1)(B+1) - \frac{B(B+1)}{2} - P] \qquad (4.3)$$

The number of terms to be evaluated in the transition matrix is much less than NT^2 as most terms turn out to be zero by observation. Let 'K' be the number of processors effectively processing at any state, then

$$K = P - (I+J) \qquad (4.4)$$

The states for a 3 × 3 × 2 system are tabulated in Figure 3.

To evaluate the transition matrix, we consider all possible transitions at time 't' into state (I,J) at time 't+Δt'. The states that we have to consider at time 't' are obviously, (I,J), (I-1,J), (K,J-1), (I+1,J) and (I,J+1). It is, of course, understood that each of these terms would apply only if the state being considered at time 't' is a valid state. The details of the evaluation of the transition matrix for Model I are given in the Appendix. The steady state probabilities, $P_{I,J}$, of all valid states (I,J) are derived. The performance is evaluated on the basis of the 3 parameters:

(1) effective processing power (EPP) = $\Sigma\ K.P_{I,J}$

(2) expected bus usage (EBU) = $\Sigma\ I.P_{I,J}$

(3) expected number of queued (EQP) = $\Sigma\ J.P_{I,J}$

V. ANALYSIS - MODEL II

Let the P × B system contain 'P' processors and 'B' buses, where $B \leqslant P$.

In Model II, the state is again defined by the doublet (I,J) with the restrictions given in equations 4.1 and 4.2. The number of states is also given by equation 4.3. The only difference is that even by observation we can see that the state (1,P-1) will never exist for a P × B system with $B > 1$ and hence its steady state probability is zero.

The transition matrix calculations for Model II are considerably more complex and the detailed evaluation is given in [8]. The performance is again evaluated on the basis of the 3 parameters EPP, EBU and EQP, as defined in Section IV.

VI. RESULTS & CONCLUSIONS

An approximate model of this type is useful in the early stages of the design of a computer system's architecture. Loosely coupled systems are becoming more cost effective and, hence, more prevalent. The initial design of such a system requires estimates of the number of processors, number of busses, and number of memory modules (if separate). Our approximate model can be used if constraints on the performance parameters, as well as, request and service statistics are estimated. The mean time for a processor to request after finishing service can be approximated knowing the average number of instructions executed before communicating with another processor and the instruction execution rate. The average message transmission time can be estimated by knowing the average message length and bus bandwidth.

The approximation is shown to be a reasonable one by comparing results of the approximate analysis with the results of the simulation of models without the approximation.

Results of both continuous and discontinuous analyses of Models I and II are plotted as functions of the number of buses for different traffic loads in Figures 4 and 5, respectively.

In both the systems the processing power is limited by three factors - 1) finite time required for communication during which the source does not do any useful computation, 2) interference due to a limited number of buses, and 3) interference due to conflicting destination references (either memories or other processors). The relative effects of these factors can be seen by evaluating the performance of the two systems with the number of buses used as a parameter. The absolute maximum performance, with no bus or destination conflicts for both these systems, is given by

$$EPP = \left(\frac{1/\lambda}{1/\lambda + 1/\lambda}\right) \cdot P \qquad [9]$$

The analysis gives good results before the knee of the graphs where the buses are the main source of conflicts. Continuous analysis is a better approximation of the system. The maximum percentage error obtained in EPP for a wide range of traffic on the bus is 7%. Discontinuous analysis tends to give errors up to 12% in EPP. The discontinuous model gives an upper bound on the performance. The nature of the approximations made in the discontinuous analysis can be intuitively shown to cause this. There are two factors in the approximation. Consider a 10 × 8 × 3 example in Model I. Similar reasonings can be given for Model II also. The memory modules being requested will be labeled by capital letters. Consider the state (2,3) which can be the result of transition from (1,3), (2,2), (3,3) or (2,4). In the state (2,3) the memory references of the three queued processors are of the form (A,A,A), (A,A,B), (A,B,A), (A,B,B), (B,A,A), (B,A,B), (B,B,A), (B,B,B).

The analysis assigns equal probability to all these possibilities. Starting with similar patterns for the source states (2,2), (3,3), (2,4) this assumption seems reasonable. The transition from state (1,3), however, will make the pattern of type (A,A,A) or (B,B,B) more likely. The amount by which the probability of these will increase will depend on the a priori probability of the state (1,3). It is obvious that trying to include corrections for this would not be trivial. Thus states with queued reference patterns of the type (A,A,A) have smaller probabilities in the analysis than in the actual system (simulation). These states have smaller probabilities of a bus being granted to a processor in the queue, due to the pattern of requests. This results in analysis giving better performance than the actual system. This reasoning is valid for both continuous and discontinuous analysis.

The other factor is the case when all the buses are being used (I=3, in the example). Continuous analysis distributes the references in the queue over a set smaller than the actual case, while discontinuous analysis distributes the references over a larger set. The larger the set the greater the chance of a grant at a service end and, hence, higher the performance. Thus discontinuous would tend to perform better than the actual system, whereas, continuous would perform poorer due to this factor.

Both factors make discontinuous performance better than the actual system. For a large number of buses the probability of states with all buses being used is reduced and, hence, continuous and discontinuous give almost the same results. Since the two factors oppose in continuous analysis, it can perform better or worse than the actual system. From the minimum error point of view, continuous analysis represents the model more accurately. The anomaly of higher performance with fewer buses in discontinuous analysis might also be attributed to the second factor discussed above. With fewer buses, the states with all the buses being used have higher probability, resulting in higher performance.

Since in reality the request and service times are not necessarily exponential, an idea of the usefulness of the analysis for non-exponential systems is necessary. The two models were simulated with the request and service fixed at $1/\lambda$ and $1/\mu$ respectively. These are superposed on the earlier graphs of EPP vs. B in Figures 4 and 5. The model with fixed request and service times performs better than the exponential case. The sensitivity to the request and service distribution is higher for lower values of EPP.

APPENDIX

The transition matrix for Model I is evaluated in this appendix. To evaluate consider all possible transitions from states (I,J), (I-1,J), (I,J-1), (I+1,J) and (I,J+1) at time 't' to state (I,J) at time 't+Δt'.

Let E_r and E_s denote events as shown below:

E_r = [Occurrence of one request and no service end in Δt]

E_s = [Occurrence of one service end and no request in Δt]

Probability of transition from (I,J) at 't' to (I,J) at

$$'t+\Delta t' = P\{\text{no arrivals or service ends in } \Delta t\}$$

$$= (1-K\lambda\Delta t)(1-I\mu\Delta t)$$

Probability of transition from (I-1,J) at 't' to (I,J) at

$$'t+\Delta t' = P\{\text{request granted} \mid E_r\} \cdot P\{E_r\}$$

$$= \left(\frac{M-I+1}{M}\right)(K+1)\lambda\Delta t[1-(I-1)\mu\Delta t]$$

the term $\left(\frac{M-I+1}{M}\right)$ is due to the memoryless averaging approximation.

Probability of transition from (I,J-1) at 't' to (I,J) at

$$'t+\Delta t' = P\{E_r\}. \; P\{\text{request not granted} \mid E_r\}$$

$$= \begin{cases} (K+1)\lambda\Delta t[1-I\mu\Delta t] & \text{if } I = B \\ (K+1)\lambda\Delta t[1-I\mu\Delta t]\left(\frac{I}{M}\right) & \text{if } I < B \end{cases}$$

Probability of transition from (I+1,J) at 't' to (I,J) at

$$'t+\Delta t' = P\{\text{none in the queue alloted} \mid E_s\} \cdot P\{E_s\}$$

$$= \left(\frac{I}{I+1}\right)^J \cdot (I+1)\mu\Delta t[1-(K-1)\lambda\Delta t] \qquad \text{if } I+1 < B$$

$$= \begin{cases} \left(\frac{I}{I+1}\right)^J (I+1)\mu\Delta t[1-(K-1)\lambda\Delta t] \text{ - continuous} \\ \left(\frac{I}{M}\right)^J (I+1)\Delta t[1-(K-1)\lambda\Delta t] \text{ - discontinuous} \end{cases} \qquad \text{if } I+1 = B$$

Probability of transition from $(I,J+1)$ at 't' to (I,J) at

$$'t+\Delta t' = P\{\text{one processor in queue allotted} \mid E_s\} \cdot P\{E_s\}$$

$$= \left[1 - \left(\frac{I-1}{I}\right)^{J+1}\right] \cdot I\mu\Delta t[1-(K-1)\lambda\Delta t] \qquad \text{if } I < B$$

$$= \begin{cases} \left[1 - \left(\frac{I-1}{I}\right)^{J+1}\right] \cdot I\mu\Delta t[1-(K-1)\lambda\Delta t] \text{ - continuous} \\ \\ \left[1 - \left(\frac{I-1}{M}\right)^{J+1}\right] \cdot I\mu\Delta t[1-(K-1)\lambda\Delta t] \text{ - discontinuous} \end{cases} \qquad \text{if } I = B$$

Setting up the probability of the state (I,J) at 't+Δt', $P_{I,J}(t+\Delta t)$ in terms of the probability of the states (I,J), $(I-1,J)$, $(I,J-1)$, $(I+1,J)$ and $(I,J+1)$ at 't', and the derived transition probabilities, and taking the limit as $\Delta t \to 0$ and equating $\frac{\partial P_{I,J}(t)}{\partial t}$ to zero we get the following relation between the steady state probabilities.

Continuous case

$$\begin{aligned} &-P_{I,J} \cdot (K\lambda+I\mu) + P_{I-1,J}(K+1)\lambda\left[\left(\frac{M-I+1}{M}\right)\right] \\ &+P_{I,J-1} \cdot (K+1)\lambda F(I,B) \\ &+P_{I+1,J} \cdot (I+1)\mu\left(\frac{I}{I+1}\right)^J \\ &+P_{I,J+1} \cdot I\mu\left[1 - \left(\frac{I-1}{I}\right)^{J+1}\right] \end{aligned} \quad = 0 \qquad (A.1.1)$$

Discontinuous case

$$\begin{aligned} &-P_{I,J} \cdot (K\lambda+I\mu) + P_{I-1,J}(K+1)\lambda\left[\frac{M-I+1}{M}\right] \\ &+P_{I,J-1} \cdot (K+1)\lambda F(I,B) \\ &+P_{I+1,J} \cdot (I+1)\mu[G(I,M,B)]^J \\ &+P_{I,J+1} \cdot I\mu[1-(G(I-1,M,B))^{J+1}] \end{aligned} \quad = 0 \qquad (A.1.2)$$

where

$$F(I) = \begin{cases} 1 & \text{if } I = B \\ I/M & \text{if } I < B \end{cases}$$

$$\text{and } G(I,M,B) = \begin{cases} (I/I+1) & \text{if } I+1 < B \\ (I/M) & \text{if } I+1 = B \end{cases} \qquad (A.1.3)$$

REFERENCES

[1] E. D. Jensen, "The Honeywell Experimental Distributed Processor - An Overview," Computer, pp. 28-37, January 1978.

[2] M. Ajmone Marsan, G. Conte, D. Del Corso and F. Gregoretti, "Architecture, communication procedures and performance evaluation of μ* multimicroprocessor system," Proceedings of the 1st International Conference on Distributed Computing Systems, pp. 106-115, October 1979.

[3] D. P. Bhandarkar, "Analysis of Memory Interference in Multiprocessors," IEEE Trans. Comput., Vol. C-24, pp. 897-908, September 1975.

[4] K. V. Sastry and R. Y. Kain, "On the Performance of Certain Multiprocessor Computer Organizations," IEEE Trans. Comput., Vol. C-24, pp. 1066-1074, November 1975.

[5] A. S. Sethi and N. Deo, "Interference in Multiprocessor Systems with Localized Memory Access Probabilities," IEEE Trans. Comput., Vol. C-28, pp. 157-163, February 1979.

[6] G. Reyling, "Performance and Control of Multiple-Microprocessor Systems," Computer Design, pp. 81-86, March 1974.

[7] D. A. Protopapas and E. J. Smith, "Modeling and Analysis of Single and Multiple Bus Multimicrocomputer System," Compcon, Fall 1980, pp. 471-478.

[8] S. Vijay Iyengar, M.S. Thesis, Department of Electrical Engineering, University of Minnesota, June 1980.

[9] L. L. Kinney and R. G. Arnold, "Analysis of a Multiprocessor System with a Shared Bus," IEEE Computer Architecture Conference, pp. 89-95, 1978.

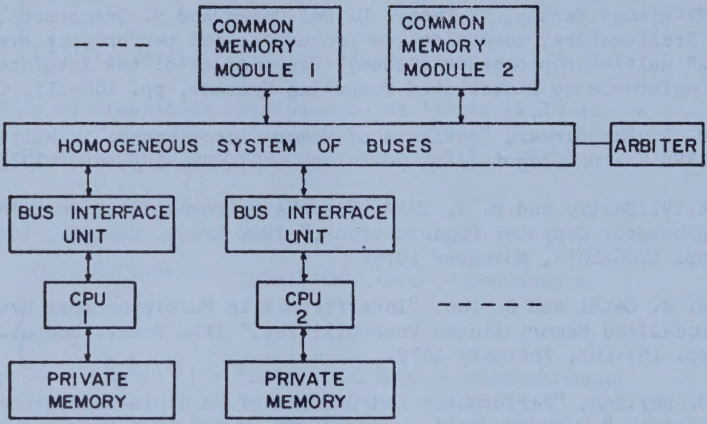

COMMON MEMORY MODULE I
COMMON MEMORY MODULE 2
HOMOGENEOUS SYSTEM OF BUSES
ARBITER
BUS INTERFACE UNIT
BUS INTERFACE UNIT
CPU 1
CPU 2
PRIVATE MEMORY
PRIVATE MEMORY

Figure 1. Block diagram of System 1.

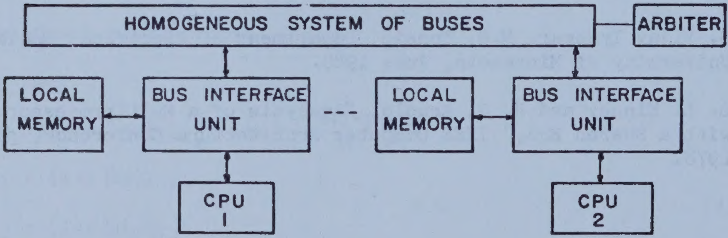

HOMOGENEOUS SYSTEM OF BUSES
ARBITER
LOCAL MEMORY
BUS INTERFACE UNIT
LOCAL MEMORY
BUS INTERFACE UNIT
CPU 1
CPU 2

Figure 2. Block diagram of System 2.

STATE	I	J	K
1	0	0	3
2	1	0	2
3	1	1	1
5	2	0	1
6	2	1	0

Figure 3. Table of states for 3 X 3 X 2 System.

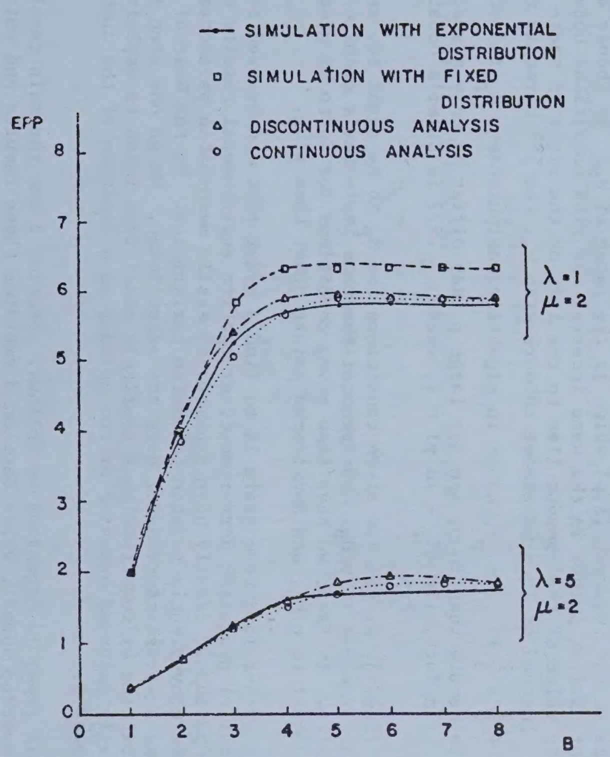


Figure 4. EPP vs B for an example of Model I with p=10, m=8.

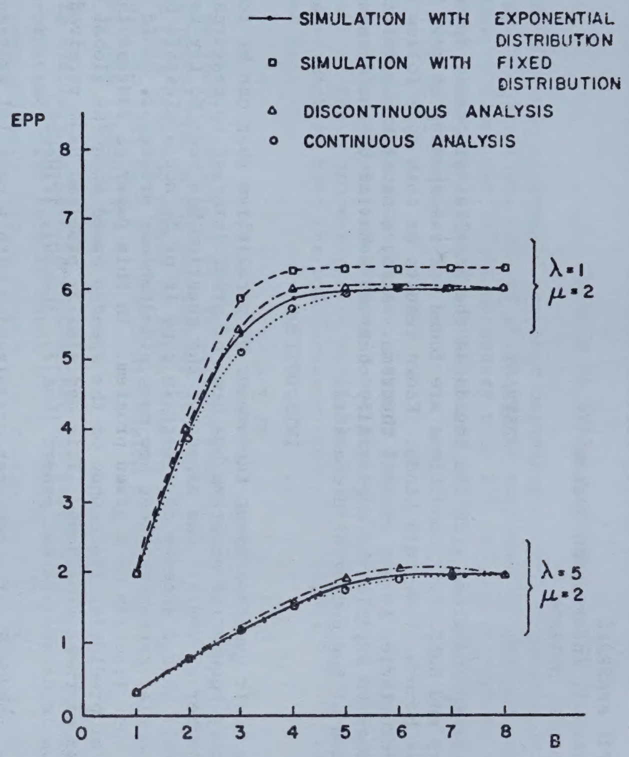


Figure 5. EBP vs B for an example of Model II with p=10.

BOUNDS ON THE PARALLEL PROCESSOR SPEEDUP

UTPAL BANERJEE
Honeywell Information Systems
Phoenix, Arizona

ABSTRACT

This paper is concerned with the bounds on the parallel processor speedup. Necessary and sufficient conditions are found for the speedup to have lower and upper bounds of certain kinds. Known results of this type follow as simple corollaries to our general theorems. Also, a heuristic model is constructed to explain the empirically observed behavior of the speedup as a function of the number of processors.

1. INTRODUCTION

The basic question about the amount of parallelism that can be found in ordinary sequential programs has been of great interest to researchers over a number of years. One aspect of the question has been to try to demonstrate that a speedup of a certain type is or is not achievable by making simple statements about how many simultaneous processors could be used on what fraction of a given program. In this paper we address the problem of predicting the bounds on the speedup based upon the global distribution of these fractions. This and similar problems have received attention in several recent papers [Lee77], [Lee80], [FlHe80].

The speedup S_p of a computer organization with p parallel processors over one with a single processor (for a given computation) is a number between 1 and p. The ideal upper bound of p is rarely achieved due to the intrinsic nature of the given computation and physical limitations of the system. It turns out in empirical observations of practical programs that the speedup is often less than p/ln p, and that p/ln p is a convenient milestone in the interval [1,p] which is the range of S_p. We choose a number of similar points in the same interval and find conditions under which the value of the speedup lies to the left or the right of these points. Instead of ln p we choose to work with H_p, the p^{th} harmonic number ($H_p = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{p}$), since in algebraic manipulations H_p is much easier to handle than ln p, and for large p they differ by only a small amount. (In fact, $\lim_{p\to\infty} (H_p - \ln p) = \gamma$, where $\gamma \simeq .577$ is Euler's constant.)

In [Lee77], R. Lee has given conditions for S_p to be equal to, and less than or equal to p/H_p. Our general theorems imply Lee's results as corollaries. In fact, we have also given conditions for S_p to be greater than or equal to p/H_p, and considered points other than p/H_p.

One of our long term goals is to find a model that can be used to predict parallel processor performance, or confirm experimental results on the same. One should ideally have a number of easily measurable parameters which are conveniently related to system performance. To the best of our knowledge a satisfactory model does not exist today. We go one step in this direction by constructing a heuristic model that tries to explain the empirically observed behavior of the speedup as a function of the number of processors.

This paper is organized as follows. Section 2 has the main results on the speedup bounds, while Section 3 explains these results and puts them in the proper perspective. The heuristic model is developed in Section 4, and that is followed by a short section giving conclusions.

2. RESULTS ON SPEEDUP BOUNDS

We consider a parallel computer organization with p processors. There is a set of operations each of which can be performed by any processor in one unit of time. It is not necessary that all processors perform the same operation in a given time slot (either the array or the multiprocessor model fits our results).

Take any given computation. Suppose that it takes T_1 time units to complete this computation when only one processor is used, and T_p units when all of the p processors are available. The speedup S_p of the p-processor machine over the uniprocessor (for this computation) is then defined by:

$$S_p = T_1/T_p .$$

Since we should have $T_p \leq T_1$, it follows that $S_p \geq 1$. Again the total number of operations in the p-processor case is $\leq pT_p$. Since this number cannot be less than the number of operations in the uniprocessor case, we get $T_1 \leq pT_p$, so that $S_p \leq p$. (For a discussion of different performance measures see [Lee80].)

To simplify matters we assume that the total number of operations in the p-processor case is also equal to T_1. (This is close to the truth for many computations.) Let q_k denote the fraction of T_1 that is executed using exactly k processors $(1 \leq k \leq p)$. Then q_k is the probability that an operation is performed at a time step when exactly k processors are active. We have $0 \leq q_k \leq 1$ for all k and $q_1 + q_2 + \ldots + q_k = 1$. Let r_k denote the fraction of T_1 that is executed using k or fewer processors $(1 \leq k \leq p)$. Then $r_k = q_1 + q_2 + \ldots + q_k$, and $0 \leq r_1 \leq r_2 \leq \ldots \leq r_p = 1$. Also, for a continuous variable x, let A(x) denote the fraction of T_1 that is executed while the number of active processors is $\leq x$. Then A(x) is a step function given by

$$A(x) = \begin{cases} r_{\lfloor x \rfloor} & \text{if} \quad 0 \leq x < p \\ 1 & \text{if} \quad p \leq x < \infty . \end{cases}$$

Finally, let H_k denote the k^{th} harmonic number, i.e.

$$H_k = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{k}, \quad (k \geq 1).$$

We first find a number of different expressions for the speedup.

LEMMA 1:

$$\begin{aligned} \frac{1}{S_p} &= \sum_{k=1}^{p} \frac{q_k}{k} \\ &= \frac{1}{p} + \sum_{k=1}^{p-1} \frac{r_k}{k(k+1)} \\ &= \frac{1}{p} + \int_1^p \frac{A(x)}{x^2}\, dx \\ &= \int_1^\infty \frac{A(x)}{x^2}\, dx \end{aligned}$$

Proof: In the p-processor calculation q_kT_1 operations are performed using k processors. Hence, the number of time steps in which exactly k processors are used is q_kT_1/k. It then follows that:

$$T_p = \sum_{k=1}^{p} \frac{q_kT_1}{k} = T_1 \sum_{1}^{p} \frac{q_k}{k} .$$

Hence $$\frac{1}{S_p} = \frac{T_p}{T_1}$$

$$= \sum_{1}^{p} \frac{q_k}{k}$$

$$= q_1 + \sum_{2}^{p} \frac{r_k - r_{k-1}}{k}$$

$$= r_1(1 - \frac{1}{2}) + r_2(\frac{1}{2} - \frac{1}{3}) + \ldots + r_{p-1}(\frac{1}{p-1} - \frac{1}{p}) + \frac{r_p}{p}$$

$$= \frac{1}{p} + \sum_{k=1}^{p-1} \frac{r_k}{k(k+1)} , \quad \text{since } r_p = 1$$

$$= \frac{1}{p} + \sum_{k=1}^{p-1} r_k \int_{k}^{k+1} \frac{dx}{x^2}$$

$$= \frac{1}{p} + \sum_{k=1}^{p-1} \int_{k}^{k+1} \frac{A(x)}{x^2} dx$$

$$= \frac{1}{p} + \int_{1}^{p} \frac{A(x)}{x^2} dx$$

$$= \int_{p}^{\infty} \frac{dx}{x^2} + \int_{1}^{p} \frac{A(x)}{x^2} dx$$

$$= \int_{1}^{\infty} \frac{A(x)}{x^2} dx$$

Q.E.D.

In this paper only the first two expressions will be used. The integral forms are interesting and may be useful in a different kind of analysis.

The following two theorems give the fundamental results on the bounds of S_p.

THEOREM 1:

If $r_k \geq ck$ $(k = 1, 2, \ldots, t)$

for some positive integer $t \leq p$ and some constant $c > 0$, then

$$S_p \leq \frac{1}{cH_t} \leq \frac{1}{c} .$$

Proof: We have

$$\begin{aligned}
\frac{1}{S_p} &= \frac{1}{p} + \sum_{k=1}^{p-1} \frac{r_k}{k(k+1)} && \text{by Lemma 1} \\
&= \frac{1}{p} + \sum_{k=1}^{t-1} \frac{r_k}{k(k+1)} + \sum_{k=t}^{p-1} \frac{r_k}{k(k+1)} \\
&\geq \frac{1}{p} + \sum_{1}^{t-1} \frac{r_k}{k(k+1)} + r_t \sum_{t}^{p-1} \frac{1}{k(k+1)} \\
&\geq \frac{1}{p} + c \sum_{1}^{t-1} \frac{1}{k+1} + ct\left(\frac{1}{t} - \frac{1}{p}\right) && \text{by hypothesis} \\
&= cH_t + (1 - ct)/p \\
&\geq cH_t \quad \text{since } 1 \geq r_t \geq ct
\end{aligned}$$

Hence $S_p \leq \frac{1}{cH_t} \leq \frac{1}{c}$. Q.E.D.

THEOREM 2:

If $r_k \leq ck$ $(k = 1, 2, \ldots, t)$

for some positive integer $t \leq p$ and some constant $c \geq \frac{1}{t}$, then

$$S_p \geq \frac{1}{cH_t} \geq \frac{1}{cH_p} .$$

Proof: We have

$$\begin{aligned}
\frac{1}{S_p} &= \frac{1}{p} + \sum_{k=1}^{t-1} \frac{r_k}{k(k+1)} + \sum_{k=t}^{p-1} \frac{r_k}{k(k+1)} && \text{by Lemma 1} \\
&\leq \frac{1}{p} + \sum_{1}^{t-1} \frac{r_k}{k(k+1)} + \sum_{t}^{p-1} \frac{1}{k(k+1)} && \text{since each } r_k \leq 1 \\
&\leq \frac{1}{p} + c \sum_{1}^{t-1} \frac{1}{k+1} + \frac{1}{t} - \frac{1}{p} && \text{by hypothesis} \\
&= c(H_t - 1) + \frac{1}{t} \\
&= cH_t - \left(c - \frac{1}{t}\right) \\
&\leq cH_t \quad \text{since } c - \frac{1}{t} \geq 0 .
\end{aligned}$$

Hence $S_p \geq \frac{1}{cH_t} \geq \frac{1}{cH_p}$. Q.E.D.

As we mentioned in the Introduction, the quantity p/H_p (the surrogate for $p/\ln p$) plays an important role in the experimental studies of the speedup. We derive next a number of conditions which force S_p to relate to p/H_p in different ways. These and similar results (to be deduced from Theorems 1 and 2) will be summarized in Section 3.

THEOREM 3:

(1) If $r_k \geq \frac{k}{p}$ $(1 \leq k \leq p)$ then $S_p \leq \frac{p}{H_p}$.

(2) If $r_k \leq \frac{k}{p}$ $(1 \leq k \leq p)$ then $S_p \geq \frac{p}{H_p}$.

(3) If $r_k = \frac{k}{p}$ $(1 \leq k \leq p)$ then $S_p = \frac{p}{H_p}$.

(4) By taking the probabilities r_k sufficiently close to $\frac{k}{p}$, we can make S_p arbitrarily close to $\frac{p}{H_p}$.

Proof: Parts (1) and (2) follow from Theorems 1 and 2 respectively, if we take $t = p$ and $c = \frac{1}{p}$. Part (3) is the joint implication of (1) and (2). We prove only Part (4). We have

$$\begin{aligned}
|S_p - \frac{p}{H_p}| &= \frac{pS_p}{H_p}\, |\frac{H_p}{p} - \frac{1}{S_p}| \\
&= \frac{pS_p}{H_p}\, |\frac{1}{p} + \sum_{k=1}^{p-1} \frac{r_k}{k(k+1)} - \frac{1}{p} \sum_{k=0}^{p-1} \frac{1}{k+1}| \\
&= \frac{pS_p}{H_p}\, |\sum_{k=1}^{p-1} \frac{r_k - k/p}{k(k+1)}| \\
&\leq \frac{pS_p}{H_p} \cdot \delta \cdot \sum_{1}^{p-1} \frac{1}{k(k+1)} \\
&= \frac{pS_p}{H_p}\, \delta\, (1 - \frac{1}{p}) \quad (\text{where } \delta = \max_{1 \leq k \leq p-1} |r_k - k/p|) \\
&\leq \frac{p(p-1)}{H_p} \cdot \delta \quad \text{since } S_p \leq p.
\end{aligned}$$

Thus, for any $\varepsilon > 0$, we can make $|S_p - \frac{p}{H_p}| < \varepsilon$ by taking $\delta < \frac{H_p}{p(p-1)}\, \varepsilon$. Q.E.D.

COROLLARY 1:

(1) If there exists a positive integer $t \leq p$ such that $q_1, q_2, \ldots q_t \geq \frac{1}{p}$ and $q_{t+1}, \ldots, q_p \leq \frac{1}{p}$, then $S_p \leq \frac{p}{H_p}$.

(2) If there exists a positive integer $t \leq p$ such that $q_1, q_2, \ldots, q_t \leq \frac{1}{p}$ and $q_{t+1}, \ldots, q_p \geq \frac{1}{p}$, then $S_p \geq \frac{p}{H_p}$.

Proof: We prove only Part (1). The idea is to show that $r_k \geq \frac{k}{p}$ $(1 \leq k \leq p)$, and then apply Theorem 3(1). We already have $r_k \geq \frac{k}{p}$ $(1 \leq k \leq t)$.

$$\begin{aligned}
\text{Now } r_{t+1} &= q_1 + q_2 + \ldots + q_{t+1} = 1 - (q_{t+2} + \ldots + q_p) \\
&\geq 1 - (p - t - 1)/p = \frac{t+1}{p}
\end{aligned}$$

Similarly, $r_{t+2} \geq \frac{t+2}{p}$, etc. Q.E.D.

COROLLARY 2:

(1) If $q_1 \geq q_2 \geq \dots \geq q_p$ then $S_p \leq \frac{p}{H_p}$.

(2) If $q_1 \leq q_2 \leq \dots \leq q_p$ then $S_p \geq \frac{p}{H_p}$.

(3) If $q_1 = q_2 = \dots = q_p$ then $S_p = \frac{p}{H_p}$.

Proof: We prove only Part (1). Since $\sum_{k=1}^{p} q_k = 1$, we cannot have $q_k < \frac{1}{p}$ for all k. Hence, there must be a positive integer $t \leq p$ such that $q_t \geq \frac{1}{p}$. Taking t to be the largest such integer, we get $q_1, q_2, \dots, q_t \geq \frac{1}{p}$ and $q_{t+1}, \dots, q_p < \frac{1}{p}$. Now use Part (1) of Corollary 1. Q.E.D.

REMARKS:

(1) Part (1) of Corollary 1 and Parts (1) and (3) of Corollary 2 were obtained by R. Lee in [Lee77].

(2) In the same paper Lee has the following Theorem:

THEOREM (Lee): $S_p \leq \frac{p}{H_p} < \frac{p}{\ln p}$ iff $\sum_{i=1}^{p} (q_i - \frac{1}{p})/i \geq 0$ ... (*). We would like to point out that this statement is slightly misleading. (*) is necessary and sufficient for $S_p \leq p/H_p$, but is not necessary and sufficient for $S_p < p/\ln p$. We have $H_p > \ln p$ and hence $p/H_p < p/\ln p$ for all integral $p > 1$. Thus if (*) holds, we have $S_p \leq p/H_p$ and consequently $S_p < p/\ln p$. However, if $S_p < p/\ln p$, then one may still have $S_p > p/H_p$ in which case (*) does not hold.

3. INTERPRETATION OF RESULTS

The speedup S_p must lie in the closed interval [1,p]. We consider a number of intermediate points of [1,p] as listed below:

$$1 < p/H_p^{\alpha} < p/H_p^{\alpha-1} < \dots < p/H_p < p/(\ln H_p)^{\beta} < \dots < p/\ln H_p < p,$$

where α is the largest positive integer such that $H_p^{\alpha} < p$, and β is the largest positive integer such that $(\ln H_p)^{\beta} < H_p$. Our goal is to show how to place the value of S_p relative to these points based on the distribution of the probabilities r_k.

It is very convenient to have a geometrical picture. (See Fig. 1.) We use rectangular axes and plot the points Q_k with coordinates (k, r_k), $1 \leq k \leq p$. All these points lie in the closed rectangle bounded by the lines $x = 1$, $x = p$, $y = 0$, and $y = 1$, since $1 \leq k \leq p$ and $0 \leq r_k \leq 1$. Note that Q_p is always the point (p,1) and that Q_{k+1} is never lower than Q_k.

The plot of the points Q_k displays graphically the amount of parallelism present in the given program. The height of Q_1 above the X-axis gives the fraction of the program consisting of serial code, the height of Q_2 above Q_1 gives the fraction executed using 2 processors, and so on. If the points Q_k rise fast then this would indicate that the program can be saturated with a small number of processors and hence there is a relatively small amount of parallelism in the program.

The rate at which the points rise can be measured qualitatively by relating their positions against straight lines passing through the origin and having different slopes. We choose a set of such lines that go through the rectangle containing the points Q_k, and whose slopes correspond to the intermediate points of the interval [1,p] described above. We then apply the theorems of the previous section to see in what part of the interval [1,p] the value of S_p lies for what kind of rate of rise for the points Q_k, and conversely.

Let L_i denote the line through the origin with slope H_p^i/p and ℓ_j the line through the origin with slope $(\ln H_p)^j/(pH_p)$, $0 \leq i \leq \alpha$, $1 \leq j \leq \beta$.

Consider first the line L_0 with slope $1/p$ that passes through the point Q_p. Any other point Q_k may lie on, above or below L_0 according as r_k is equal to, greater than, or less than k/p, respectively. How the value of S_p relates to p/H_p depends to a great extent on the positions of the points Q_k relative to L_0, as shown by Theorem 3. Each line L_i with $1 \leq i \leq \alpha$ is such that not all points Q_k may lie on or above it. And each ℓ_j with $1 \leq j \leq \beta$ is such that not all Q_k may lie on or below it. The Theorems of the previous section may be used to generate results about the position of the value of S_p in [1,p], based on the distribution of the points Q_k relative to these lines. For example, consider the line L_2 with slope H_p^2/p. Taking $c = H_p^2/p$ and $t = 1$ in Theorem 1, we see that $S_p \leq p/H_p^2$ when Q_1 lies on or above L_2. And taking $c = H_p^2/p$ and $t = p$ in Theorem 2, we get that $S_p \geq p/H_p^3$ when all points Q_k lie on or below L_2. These results are summarized in Tables 1 and 2.

REMARKS

(1) Theorem 2 cannot be used for $c < 1/t$. Hence $c = (\ln H_p)^j/(pH_p)$ is not acceptable and we have no results on the bounds of S_p when some of the points Q_k lie below a line ℓ_j.

(2) Roughly speaking, the more slowly the points Q_k rise the greater is the value of the speedup S_p.

(3) The analysis we have given provides a nice theoretical framework for further experimental work. However, one should be a little careful in relating experimental results to theory. For example, it follows from the work of Kuck and others [Kuck74], [Lee80] on the University of Illinois Program Analyzer that $S_p \leq p/H_p$ for a large percentage of Fortran programs. From this we cannot conclude that all points Q_k must lie on or above the line L_0 (see Table 2). The probabilities r_k may be distributed quite randomly and still make $S_p \leq p/H_p$.

(4) It is not hard to show that p/H_p is an increasing function of p, diverging to ∞. Since S_p has an absolute upper bound of T_1 (independent of p), it can always be made $\leq p/H_p$ if we choose p large enough. For any given computation saturation is reached at some point P such that $r_P = r_{P+1} = \ldots = r_p = 1$ for any $p > P$. If we increase p beyond P, we increase p/H_p, but S_p remains constant as the extra processors cannot be put to use.

(5) Since all the probabilities q_k are nonnegative it follows from Lemma 1 that k/q_k is an upper bound on S_p for any k such that $q_k > 0$. In particular, if the given program contains some inherently serial code, then $q_1 > 0$ and we have $S_p < 1/q_1$. However, this result is not always very interesting, since for many computations q_1 is close to zero and hence $1/q_1$ is a very large number.

4. AN EMPIRICAL MODEL

The speedup of a computation on a p-processor machine can be exactly

IF	THEN
1. Each Q_k lies on or above L_0	$S_p \le p/H_p$
2. Each Q_k lies on or below L_0	$S_p \ge p/H_p$
3. Each Q_k lies on L_0	$S_p = p/H_p$
4. Q_1 lies on or above L_i	$S_p \le p/H_p^i$
5. Each Q_k lies on or below L_i	$S_p \ge p/H_p^{i+1}$
6. Each Q_k lies on or above ℓ_j	$S_p \le p/(\ln H_p)^j$

$(1 \le i \le \alpha,\ 1 \le j \le \beta)$

TABLE 1

IF	THEN
1. $S_p < p/H_p$	At least one Q_k lies above L_0.
2. $S_p > p/H_p$	At least one Q_k lies below L_0.
3. $S_p \neq p/H_p$	At least one Q_k is not on L_0.
4. $S_p < p/H_p^{i+1}$	At least one Q_k lies above L_i.
5. $S_p > p/H_p^i$	Q_1 lies below L_i.
6. $S_p > p/(\ln H_p)^j$	At least one Q_k lies below ℓ_j.

$(1 \le i \le \alpha,\ 1 \le j \le \beta)$

TABLE 2

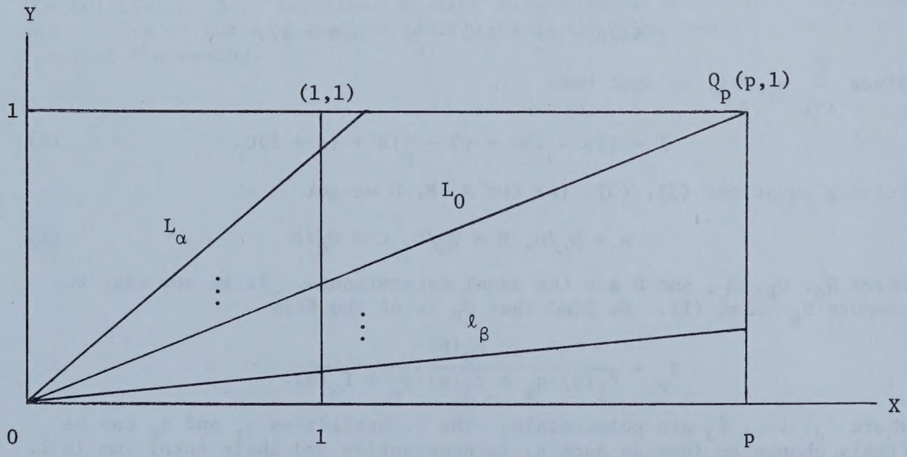


FIG. 1

represented by considering the entire set of probabilities $\{q_1, q_2, \ldots, q_p\}$. We seek a simple model that 1) contains an expression for q_i such that the distribution of q_1 as a function of i is similar to the experimentally observed distributions, and 2) yields an overall speedup that is similar to empirical speedup values. Such a model will typically be applied when only a limited number of parallel processors are being used. For real computations it is reasonable to assume that $p \ll T_1$. And when there are lots of big array and recurrence operations one gets a spike at p, i.e. q_p is rather large.

The following model satisfies the above constraints intuitively. It allows large values of q_1 and q_p that can be independently chosen. It then has non-zero q_i values only at power-of-two values of i. (p is assumed to be a power of 2.) We may choose a spike at q_1 to account for all serial code (which is sometimes significant), and a spike at q_p to account for all array operations as well as parts of recurrences folded onto a fixed size machine. The other non-zero q_i's can be viewed as representing those parts of recurrences that reduce arrays to scalers.

Assume that p is a power of 2. Let the q_i's be defined by

$$q_i = \begin{cases} Ai + \frac{B}{i} + C & \text{if } i = 1, 2, 2^2, \ldots, 2^n \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $n = \log_2 p$ and A, B, C are constants. We will first express A, B, C in terms of q_1, q_p. Using (1) we get

$$q_1 = A + B + C \tag{2}$$

and

$$q_p = Ap + \frac{B}{p} + C \tag{3}$$

Now

$$\begin{aligned}\sum_{i=1}^{p} q_i &= \sum_{j=0}^{n} q_{2^j} = A \sum_{j=0}^{n} 2^j + B \sum_{j=0}^{n} 2^{-j} + C \sum_{j=0}^{n} 1 \\ &= A(2^{n+1} - 1) + B(2 - 2^{-n}) + C(n + 1) \\ &= A(2p - 1) + B(2 - \frac{1}{p}) + C(n + 1).\end{aligned}$$

Since $\sum_{i=1}^{p} q_i = 1$, we must have

$$1 = (2p - 1)A + (2 - \frac{1}{p})B + (n + 1)C. \tag{4}$$

Solving equations (2), (3), (4) for A, B, C we get

$$A = D_A/D, \; B = D_B/D, \; C = D_C/D \tag{5}$$

where D_A, D_B, D_C, and D are the usual determinants. It is now easy to compute S_p using (1). We find that S_p is of the form

$$S_p = \frac{f_0(p)}{f_1(p) \cdot q_1 + f_2(p) \cdot q_p + f_3(p)}$$

where $f_0, \ldots, f_3$ are polynomials. The probabilities q_1 and q_p can be freely chosen as long as each q_i is nonnegative and their total sum is 1.

In preliminary studies, good correlation is seen between our model and sets of real FORTRAN programs. A number of key measurements are to be made in the near future by the Computer Organization Group at the University of Illinois, and we expect to give an experimental validation of this model in a subsequent report.

5. CONCLUSIONS

We have tried to relate the value of the parallel processor speedup to the distribution of probabilities for processor usage. Also, a simple empirical model has been constructed to match the observed values of the speedup with the observed distributions of these probabilities. It would be very interesting if one could make general statements about the probabilities for a given type of computation, and then theoretically predict a range of values for the speedup (based on the results of Section 2) which is subsequently confirmed by observed statistics.

REFERENCES

[Amda67] Amdahl, G. M., "Validity of the Single Processor Approach to Achieving Large Sacle Computing Capabilities", AFIPS Conf. Proc., vol. 30, 1967, pp. 483-485.

[Bane79] Banerjee, U., "Speedup of Ordinary Programs", Ph.D. Thesis, University of Illinois at U-C, Department of Computer Science Report No. 79-989, Oct. 1979.

[Chen71] Chen, T. C., "Unconventional Superspeed Computer Systems", AFIPS Conf. Proc., vol. 38, 1971, pp. 365-371.

[FlHe80] Flynn, M. J., and Hennessy, J. L., "Parallelism and Representation Problems in Distributed Systems", IEEE Trans. Comput., vol. C-29, No. 12, Dec. 1980, pp. 1080-1086.

[Kuck73] Kuck, D. J., et al, "Measurements of Parallelism in Ordinary FORTRAN Programs:, Proc. 1972 Sagamore Comput. Conf. on Parallel Processing.

[Lee77] Lee, R. B., "Performance Bounds in Parallel Processor Organizations". Proc. Symp. on High Speed Comp. and Alg. Organization, Academic, NY, 1977.

[Lee80] Lee, R. B., "Empirical Results on the Speed, Efficiency, Redundancy and Quality of Parallel Computations", Proc. 1980 International Conf. on Parallel Processing.

CENTRALLY CONTROLLED SEGMENTED BUS ARCHITECTURE

E.T. FATHI and M. KRIEGER
Department of Electrical Engineering, University of Ottawa, Canada

ABSTRACT

One of the most important requirements of a multi-microcomputer system is to provide interprocessor communication with a high degree of availability and reliability. In this paper, the segmented bus concept, which can accomodate simultaneous mutually exclusive transfers with minimal modifications to a standard bus, is introduced. The principle of the segmented bus is that two communicating elements require only that portion of the bus which is physically between them, and this section can be isolated by inserting bidirectional switches between adjacent elements on the bus. This will permit the remaining sections of the bus to be used for other transfers. To allow for orderly communication on a segmented bus, the switches can be controlled either centrally or in a distributed fashion. Here we shall consider the centrally controlled scheme as it provides a more flexible system.

INTRODUCTION

In a multiple processor system the interprocessor communication scheme should be chosen such as to optimize system performance in terms of system throughput. Factors which should be considered when deciding on the bus organization are: flexibility, expandibility, availability, reliability and cost effectiveness.

The fully connected interconnection scheme, which provides a direct point-to-point connection, is a conflict free busing scheme. This option is not practical for large systems, since to provide complete interconnection among N elements, N(N-1)/2 connections are required. Furthermore, the control aspect of such an interconnection scheme increases in complexity as N increases.

In order to reduce the number of interconnections yet maintain an effective throughput on the bus, some kind of bus sharing must be used. However, once any type of shared bus is considered, one cannot guarantee conflict free communication between the various elements. Note that in this context an element may represent a complete processing unit and/or a resource with its associated controller. The two main bus sharing schemes are: the time shared bus and the line multiplexed bus.

In any time sharing scheme, there is generally a single resource which is being shared among the different elements of the system, on the basis of a time slot assignment. The two basic time sharing protocols are:

(a) Synchronous - where each element is given a fixed time slot during which it has the sole use of the resource. If the element has no need for the resource, its time slot is "wasted".

(b) Asynchronous - only the element that has a need requests the use of the resource. When a request is granted, it can be either for the duration of the message or for only a portion of the message. Note that a more elaborate control is required for implementing this option, as some control information must be sent in order to identify message originator and message length.

The time-shared bus, as it uses a single resource, is relatively inexpensive, but as the number of elements increases, so does the complexity of the control scheme. The time-shared bus has two major drawbacks:

(i) contention increases rapidly with the number of elements and,

(ii) since it is a single resource system, it is very susceptible to failure.

In a line multiplexed bus, separate physical connection between any two elements is via a switching system. Networks utilizing this scheme have high throughput as they can support simultaneous, mutually exclusive transfers. The most complex part of the communication system is the switching unit which provides the required connections. The major drawback of this scheme is the cost and the susceptibility of the switching system to failure.

In practice, there are a large number of busing schemes which are various combinarions of time and line sharing. Further details about these schemes can be found in [1], [2], [3].

THE SEGMENTED BUS CONCEPT

In general the motivation behind the various busing schemes is to find a structure which can be easily implemented, be cost effective, and optimize system performance. Here, a method of upgrading a standard single bus with minimal change to accomodate simultaneous, mutually exclusive communication is presented. The principle of the segmented bus is that two communicating elements only require that portion of the bus physically between them. This will permit the remaining sections of the bus to be used for other communication. Freeing the "unused" sections of the bus is accomplished by inserting bidirectional switches between adjacent elements on the bus. The ability to segment the bus into separate sections led to the term "Segmented Bus" [4].

The organization of an open ended segmented bus (OESB) is shown in Figure 1. During communication, by using the bidirectional switches, the bus can be effectively divided into a maximum of $N/2$ segments, (N = Even). This maximum value occurs only if all requests are between adjacent elements and are mutually exclusive. Note that in this context a link is defined as the direct physical distance between two adjacent elements, whereas a path or segment may consist of one or more links.

The OESB does not provide maximum resource utilization. Therefore, if the physical distances are not large, by connecting both ends of the bus via a bidirectional switch, one can obtain better resource utilization. The new organization obtained is called the Closed Loop Segmented Bus (CLSB) and is shown in Figure 2. In this organization each element controls the nearest switch to its clockwise (CW) direction. Signalling information is transmitted and received via dedicated signalling lines connecting each two adjacent elements.

In the case of the segmented bus, it makes sense to define distance between two elements as the number of intermediate nodes. The distance between element i and element j for OESB is given by $|i-j|-1$ and for CLSB by $\min[(\pm|i-j|)\mathrm{MOD}_n-1]$.

Although the segmented bus does not guarantee conflict free communication, it does reduce the total number of possible conflicts in a time-shared bus. A detailed presentation of the probabilistic nature of the segmented bus is given in [5]. Requests for communication can be checked for conflicts by comparing their source and destination specifications. Consider two communication pairs P_1 (ij) and P_2 (kl) where $i < j$, and $k < l$. Without loss of generality, it can be assumed that $i < k$. A conflict will occur in the OESB when $k < j$ and in the CLSB when $k < j < l$. The closed bound for the CLSB implies that it is less subject to conflicts than the OESB.

In order to provide controlled communication on a segmented bus, the following basic control information must be provided with each transaction: request initiation, source and destination codes, and start and end of transmission information. The various control functions associated with the segmented bus can be done either centrally or in a distributed fashion. In a distributed control segmented bus (DCSB), each element is responsible for all control functions associated with the interprocessor communication. The distributed control scheme can be implemented with various levels of complexity, In the most simplistic form each element has a hardwired bus interface which provides for a very rigid but fast interprocessor communication. A communication protocol suitable for DCSB and the drawbacks of this architecture are presented in [6].

CENTRALLY CONTROLLED SEGMENTED BUS ARCHITECTURE

In the case of short, physical distances between elements, a centralized control may be more appropriate. As opposed to the DCSB, in a centrally controlled segmented bus (CCSB) all the interconnections are controlled from a central location. The basic CCSB architecture consists of two layers in the form of rings: the outer ring is occupied by the data layer whereas the inner ring is occupied by the control layer. The data layer consists of a closed loop segmented data bus. The various elements, excluding the bus controller, are connected to the bus via bidirectional switches. Each element has one set of bidirectional switches in its physical space. The control layer consists of a closed loop control bus and some dedicated control lines. The various elements are attached to the control bus via individual control interface (CI) units. The bus controller has its own dedicated interface card which is connected both to the control bus and to all the control interface units in the system. A simple block diagram of the CCSB architecture is shown in Figure 3.

The most important element of the CCSB architecture is the central bus controller which performs all the control functions associated with the interprocessor communications in the system. The major responsibilities of the central bus controller are:

Communication functions:

(a) Accepts requests for communication from all the elements in the system.
(b) For fast messages (under 10 bytes) or in the case of heavy traffic and/or link failure, acts as mail delivery service.
(c) In the case of element and/or link failure, advises the other elements in the system directly affected to switch to their stand alone routines.

Arbitration functions:

(a) Processes new requests, taking into account their software settable and hardware communication priorities.
(b) Rearranges existing paths (if possible) to accomodate new requests.
(c) Finds the shortest path from source to destination.
(d) Maintains and supervises system communication access control by checking request validity (a request for access to a classified data base may be rejected).

Allocation functions:

(a) Assigns communication paths by physically controlling the bidirectional switch control lines.

Diagnostic functions:

(a) Utilizes a time-out option on every communication link for detecting hardware failure.
(b) Checks unused communication paths to verify that they are operational.

General chores:

(a) Keeps a real-time clock information and passes it upon request to other elements in the system.
(b) Gathers statistical data about bus controller and system operation.

Due to the large and diverse number of functions the bus controller has to perform, and the fact that it is done by a single entity, makes the bus controller the most critical element in the system. This in turn makes the CCSB architecture vulnerable to failures as a failure of the bus controller may lead to complete system failure. To reduce the probability of major system failure, the central bus controller must be designed and implemented in the most reliable fashion possible. One possible solution would be to implement the bus controller as a task-driven multi-microcomputer system [7], as outlined in [6].

A possible communication protocol for the CCSB architecture is presented below in a point form.

(1) Any element may issue a request via a dedicated request line through the control interface. Prior to this, the requesting element places in the control interface the following control information: source and destination(s)

codes, estimated length of message, and type of transaction.
(2) The central controller reads the information in the control interface of the highest priority element and clears its flag. This allows for simultaneous requests.
(3) The central controller continues to read, in order of their priority, all pending requests.
(4) Meanwhile, the central controller processes stored requests and decides on proper routing schemes.
(5) Once the controller determines the route, it signals the source and destination elements and sets the proper switches on the segmented data bus. It also updates the routing tables.
(6) The central controller sets a byte counter corresponding to the maximum message length to protect against:
- (i) Switch/link/element failure.
- (ii) Inefficient use of the bus.

(7) If a high priority element requests use of an occupied resource or bus, the controller may execute one of the following:
- (i) Revoke the existing communication links and grant the link to the high priority element.
- (ii) If the timer indicates that the message is about to be completed soon, the controller may wait for the end of the message and then grant use of the bus to the high priority element.
- (iii) The controller may act as a mail delivery service for the high priority element by accepting the message and passing it to/from the proper destination.

IMPLEMENTATION ASPECTS

The three main entities of the CCSB architecture are the data and control buses, the central bus controller and its interface to the system, and the control interface of individual elements. In the following, some of the most important design and implementation aspects as related to each of the above entities will be described.

a. Data and control buses. There are two main buses in the CCSB architecture: a data bus which is dedicated entirely to transfer of data, and a control bus which may carry both control information and data. The data bus is a closed loop segmented bus, where the bidirectional switches of the segmented bus could be a set of two tristate buffers connected in a back to back fashion. The normal switch position is the off position. Whenever there is a need for communication, the proper switches are turned on. The bidirectional switches are controlled, via two dedicated control lines, directly by the central bus controller. The two control lines are needed in order to support three possible conditions of the bus: CW path, CCW path and element isolation.

The various states of the two control lines can be used for detecting improper switch setting. Thus before any data exchange, the communicating elements verify that the switches are set properly by checking the status of the appropriate switches.

The control bus is also a closed loop bus, to provide an alternate path in the event of an open circuit, and also supports bidirectional data flow. A secondary function of the control bus is for mail delivery service, which is performed either in the cases of high priority short messages, broadcasting of a message, or in the event of a partial failure of the data bus.

b. Central bus controller and its interface. The various responsibilities of the central bus controller were outlined previously. Here, only the aspects of element request identification and the access into the buffer memory of a particular control interface are considered. These aspects are very important since the initiation of asynchronous requests by multiple elements in a system may lead to a processing bottleneck, thus lengthen the response time to a request. In order to overcome this, a fast request identification procedure is implemented completely in hardware. In order to introduce some flexibility each control interface is assigned a software settable priority.

In the proposed design, each control interface has two dedicated control lines which are connected directly to the bus controller. The first is the request line which is directed from the control interface towards the bus controller. The request line is hardwired, based upon a predetermined fixed priority, to a priority encoder latch whose outputs are expanded by a decoder. The second is the clear request line which is one of the outputs of the decoder, and is directed from the bus controller towards the element interface.

The circuit operates in the following manner. The status of the various request lines are continuously sampled. As soon as a request is initiated, the clock is inhibited and the appropriate clear request line changes its state to a logical low. In the event of multiple simultaneous requests, the highest priority request is latched. Upon detection of the request initiation, the bus controller is interrupted and subsequently reads the proper information from the memory selected by the clear request line. Upon completion of the read cycle, the clock is re-enabled so as new requests may be loaded into the priority encoder.

In order to allow the bus controller to access the various memories in different control interfaces on its own initiative, it can force one or more clear request lines into a particular state. A block diagram of the hardwired priority encoder and the interface handshake control is shown in Figure 4. The proposed design minimizes the requesting element identification time thus reducing the overall response time to a request.

For added flexibility, in addition to the fixed priority, each element is assigned a software settable priority which may be different and always takes precedence over it. The various priorities are stored in the form of a look-up table which is managed by the bus controller. In general, the hardwired priority is utilized primarily for the process of request identification. The software settable priority on the other hand is used for arbitration when deciding which elements are to be granted communication paths. Finally, it should be noted that the utilization of a software settable priority facilitates the fine tuning of the overall system performance.

c. Element control interface. The transfer of control information between each element and the central bus controller is done via a control interface whose block diagram is shown in Figure 5. The central element of this interface is a 16 byte read/write memory. An element requesting service places the proper control information, as defined in the protocol, at locations 0 to 3 of this memory. The rest of this memory is used as a "mail box" when the controller acts as a mail delivery service or when suspending communication paths.

Under normal conditions, the central controller, upon detecting a request, is given automatic access to the memory in the interference circuit of the requesting elements. The controller maintains control until it completes reading the control information from locations 0 to 3. At the end of the read cycle, control is returned to the element so that it can initiate other requests.

Whenever the bus controller wishes to deliver mail to an element or to suspend established communication paths, it has to perform the following steps:

(1) Inhibit the clock to the priority encoder latch.
(2) Record the current status of any request lines for those elements involved in the transaction. The number of elements involved in one transaction may range from one for the case of delivering a fast message, to the entire set of elements as in the case of a message broadcasting.
(3) Tristate all clear request lines.

(4) Force to logical zero a clear request line corresponding to one of the elements involved in the transaction. This clears the request latch and permits the bus controller to access the memory of the appropriate element.
(5) If a request was initiated, the bus controller reads the request information from locations 0 to 3.
(6) If the request was not initiated or after executing Step 5, the bus controller writes the proper control information (read, write, type of information e.g. suspension of communication) in locations 4 and 5.
(7) When delivering mail, the bus controller writes into or reads from the appropriate memory locations. If performing suspension of communication, it skips this step.
(8) Enable momentarily the clock to the priority encoder latch.
(9) Record for the remaining elements involved in the transaction, the status of their request lines to verify if new requests were initiated. This is necessary in order to maintain control information integrity and to avoid losing requests, as when accessing the memory the request latch is cleared.
(10) Relinquish control over the memory by re-enabling the clear request lines.
(11) Repeat steps 3 to 10 inclusive until all elements involved in the transaction are reached and are informed. This is done to maintain data integrity and control information integrity.

If the controller is to receive "mail" from an element, a regular request is transmitted. The only difference is that after reading the control information (which indicates this) the controller continues to read data from locations 6 to 15. With the proposed implementation, the controller has the ability to broadcast a message simultaneously to all or selective groups of elements.

Each time the bus controller accesses the control interface memory, an interrupt is generated, which indicates to the element that a transaction in which it is involved took place. Following an interrupt, the element reads the control information in positions 4 and 5. If it is to receive mail, it reads the data from locations 6 to 15. If it is to transmit data, it places the requested data in locations 6 to 15 and uses the normal request procedure to indicate this to the central controller. If the control information indicated suspension of a communication path, the element takes the proper action.

In order to reduce the number of dedicated direct control lines, the clear request line has a dual function: memory access enable, and element interrupt.

Each control interface also includes a simple error detection logic for checking switch status, which consists of an exclusive OR gate. The element, prior to data transaction, may read the output of the gate to find out whether the switch is in isolation mode. However, using this simple circuit, the element cannot find out whether the switch is in the forbidden state and/or the actual flow direction of data allowed by the switch. In order to get the complete picture, the element must examine the control lines of both the adjacent switch as well as the switch located in its physical space, prior to data transaction. In the event that the switch setting is not in agreement with the requested setting, it can inform the bus controller to check the switch status. This in effect also provides an indirect way of checking the operation of the bus controller.

CONCLUSIONS

The segmented bus allows for multiple, mutually exclusive communication to take place simultaneously, thus upgrading the capabilities of a single bus. The improvement in resource utilization is achieved by isolating a "busy" section on the bus and using the rest of the bus for other communication. The segmented bus has a fail soft capability since a single link failure need not be a complete system failure. By isolating a faulty part from the system, the system can still operate at reduced capacity. To improve communication and reduce contention, elements with heavy communication activities are placed in close proximity to each other.

The suggested CCSB organization has the best properties of the segmented bus and eliminates some of the drawbacks of the DCSB organization at added complexity. The CCSB enhances the fail soft capability of the segmented bus to a limited fail safe capability. Time-out option and diagnostics routines can be used to detect a faulty switch/link. An alternate data path is provided for each element via the control/data bus. This alternate path may be used when a switch/link failure is detected. In addition, for extremely high reliability, one can allow the transformation of CCSB architecture into a DCSB architecture. Thus, by properly modifying the control interface, the centrally controlled organization can be switched to distributed control in the case of controller failure.

Generally, the CCSB is best suited for highly interactive networks with short physical distances among its nodes. In the case of large distances, the use of separate control/data bus and additional control lines may not be as cost effective. A network with large numbers of elements may require an additional segmented data bus, or dedicated data bus if there are heavy communication activities. The option of an additional dedicated data bus may be used, as the switches along the data bus increase the total propagation delay through the communication path.

The CCSB organization has the advantage that bus controller capability can be enhanced without disturbing the rest of the system. In the first phase of implementation, it is good practice to provide all the necessary hardware connections from the elements to the bus controller. It is not essential to include all the functions described for the bus controller during this phase. Only the functions, specified in terms of tasks which are vital to control and maintenance of basic communication, must be implemented. As the bus controller design is based on a multi-microcomputer architecture the remaining functions may be added in later stages of the development, or as required.

Finally, from a cost point of view, the DCSB organization can be regarded at about the same level as a time shared bus but with improved communication. The CCSB organization can be compared to the crossbar switch scheme. Although the CCSB organization cannot provide the same system throughput as the crossbar switch, it has other features which cannot be implemented by the crossbar switch. Thus the cost of the CCSB organization is a direct function of how many of the bus controller functions are implemented.

REFERENCES

[1] C. Weitzman, "Distributed Micro/Minicomputer Systems, Structure, Implementation and Applications", Prentice-Hall, Inc., Englewood Cliffs, N.J. U.S.A., 1980.

[2] G.A. Anderson and E.D. Jensen, "Computer Interconnection Structures: Taxonomy Characteristics and Examples", ACM Computing Surveys, Vol. 7, No. 4, Dec. 1975, pp. 197-214.

[3] K.J. Thurber et al., "A Systematic Approach to the Design of Digital Busing Structures", Proc. IEEE Fall Joint Computer Conf. 1972, pp. 719-740.

[4] P.M. Thompson and Z.H.Glanz, "A Data Sorting System Using High Speed Bus", AFIPS Conference proceeding 1975.

[5] J.W.E. Davidson, "The Design of Fault Tolerant Interconnection Networks", Ph.D. Thesis, University of Illinois, Urbana-Champaign, U.S.A., 1978.

[6] E.T. Fathi, "Task-driven Multi-microcomputer System", M.A.Sc. Thesis, University of Ottawa, 1981.

[7] M. Krieger, "Task-driven Multi-microprocessor System", Proceedings of the First Canadian Workshop on the Design and Development of Computer Systems, May 1979, pp. 81-88.

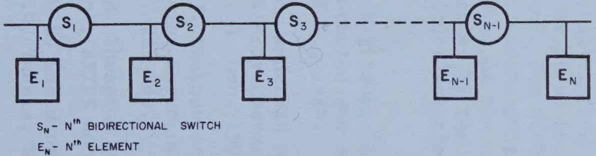

S_1
S_2
S_3
S_{N-1}
E_1
E_2
E_3
E_{N-1}
E_N
S_N - Nth BIDIRECTIONAL SWITCH
E_N - Nth ELEMENT

FIGURE 1-OPEN ENDED SEGMENTED BUS

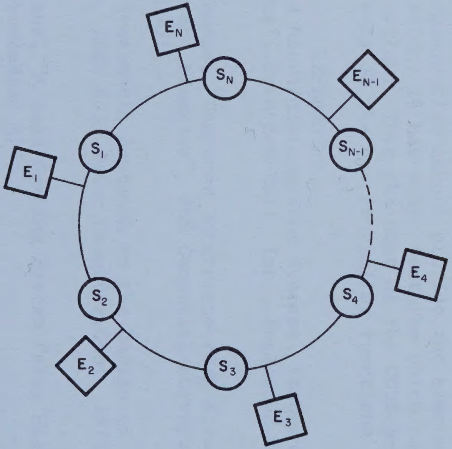

E_N
S_N
E_{N-1}
S_{N-1}
E_1
S_1
E_4
S_4
S_2
E_2
S_3
E_3

FIGURE 2-CLOSED LOOP SEGMENTED BUS

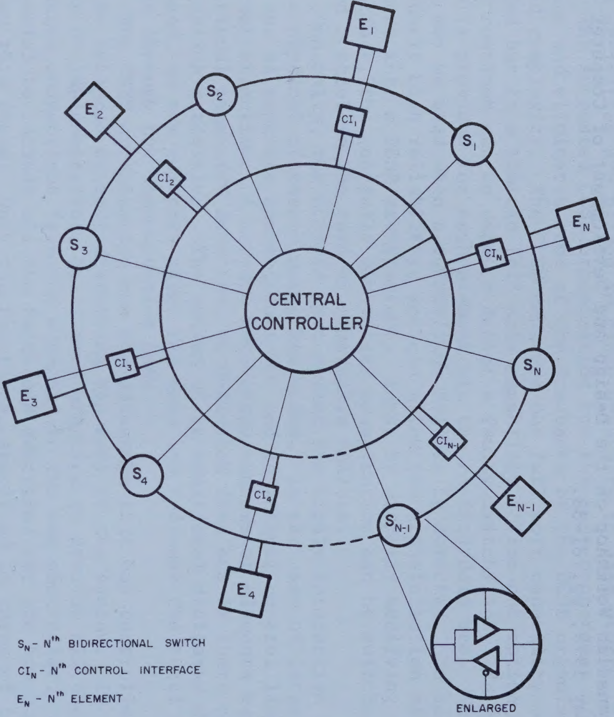

E_1
S_2
E_2
CI_1
S_1
CI_2
E_N
S_3
CI_N
CENTRAL CONTROLLER
CI_3
E_3
S_N
CI_{N-1}
S_4
CI_4
E_{N-1}
S_{N-1}
E_4
S_N - Nth BIDIRECTIONAL SWITCH
CI_N - Nth CONTROL INTERFACE
E_N - Nth ELEMENT
ENLARGED

FIGURE 3-CENTRALLY CONTROLLED SEGMENTED BUS

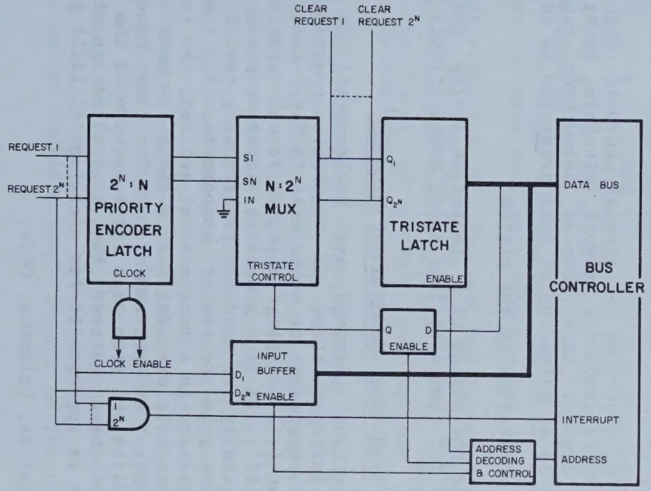

CLEAR REQUEST 1
CLEAR REQUEST 2^N
REQUEST 1
REQUEST 2^N
2^N : N PRIORITY ENCODER LATCH
CLOCK
CLOCK ENABLE
S1
SN
IN
N : 2^N MUX
TRISTATE CONTROL
Q_1
Q_{2^N}
TRISTATE LATCH
ENABLE
Q D ENABLE
INPUT BUFFER
D_1
D_{2^N} ENABLE
1
2^N
DATA BUS
BUS CONTROLLER
INTERRUPT
ADDRESS
ADDRESS DECODING & CONTROL

FIGURE 4- HARDWIRED PRIORITY ENCODER & INTERFACE HANDSHAKE CONTROL

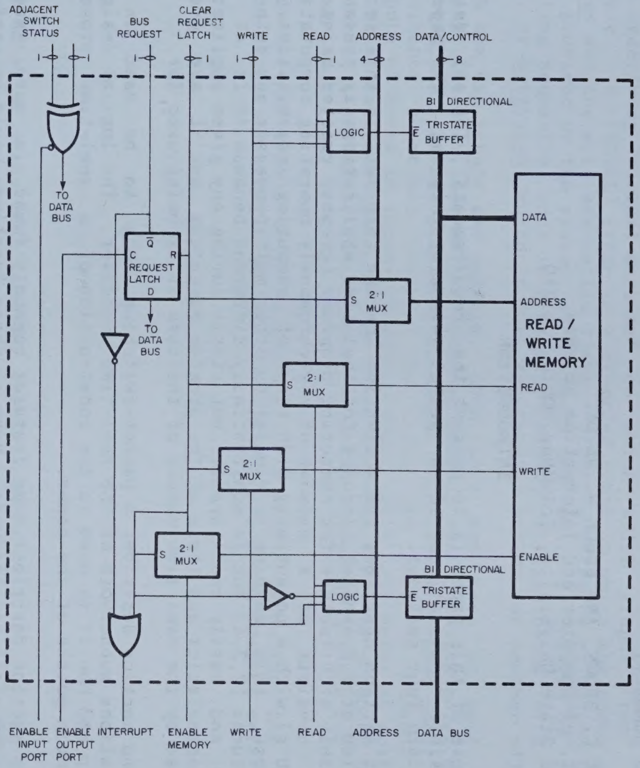

ADJACENT SWITCH STATUS
BUS REQUEST
CLEAR REQUEST LATCH
WRITE
READ
ADDRESS
DATA/CONTROL
1
1
1
1
1
1
4
8
TO DATA BUS
LOGIC
BI DIRECTIONAL
TRISTATE BUFFER
E
Q̄
C
R
REQUEST LATCH
D
TO DATA BUS
S 2:1 MUX
S 2:1 MUX
S 2:1 MUX
S 2:1 MUX
LOGIC
BI DIRECTIONAL
TRISTATE BUFFER
E
DATA
ADDRESS
READ / WRITE MEMORY
READ
WRITE
ENABLE
ENABLE INPUT PORT
ENABLE OUTPUT PORT
INTERRUPT
ENABLE MEMORY
WRITE
READ
ADDRESS
DATA BUS

FIGURE 5-CONTROL INTERFACE BLOCK DIAGRAM

A FLEXIBLE PACKET-SWITCHING CHIP SET AND ITS APPLICATIONS

(Extended Summary)

MARK E. BROWN* AND BRUCE W. WEIDE**
Dept. of Computer and Information Science,
Ohio State University, Columbus, Ohio 43210

INTRODUCTION

The purpose of this paper is to present the requirements, logical design, and possible applications of a packet-switching chip set. The paper is divided into four sections.

The first section describes the motivation for the chip set. In this section an overview of a graphical language for real-time applications is presented. One possible architecture for executing programs (graphs) created using this approach consists of a network of asynchronously operating computational units (CU's) with a potentially wide range of computing capabilities. In some sense, this machine is similar to other multiprocessor and data-flow architectures [2,3,4] but is substantially different because it is at once flexible and easily configurable, but static during any given application. Its characteristics and those of the program executing on it are greatly influenced by the real-time nature of the task it is being used for.

The second section describes a packet-switching chip to be used in the communications subnetwork of the real-time processor. The logical design of this chip and how it is used in the construction of a real-time processor are the main points of the paper.

The third section describes some features commonly found in more general-purpose communications hardware. Many of these functions would be useful in a general purpose chip.

Finally, the last section describes a general purpose chip set for communications. This set would contain a Network Interface, Packet-Switcher (PS), and add on memory to support more general networking capabilities. The packet-switcher design is an extension of that presented in section 2.

MOTIVATION OF THE DESIGN

The original motivation for the design of a packet-switching chip was to provide a packet-switcher for communications among computational units in a real-time processor [1,12].

A System for Constructing Real-Time Application Programs

STILE (STructure Interconnection Language and Environment) is a system designed to facilitate interconnection of reuseable program parts and, furthermore, to permit the programmer or other user to "see" exactly how the parts interact and to take full advantage of human visual capabilities that are unmatched by machine [12]. It is a graphical interconnection language [2,8,9,10] which was originally intended for constructing real-time process control application programs from reuseable components [1]. Because of STILE's apparent applicability in a more general setting, the semantics of the graphical language components are defined without regard for their intended application to real-time programming. On the other hand, since we will use the real-time application area as a first example of the power and convenience of STILE, we have pursued the logical design of a special "real-time" processor that is appropriate for execution of STILE programs.

---

* Also with Bell Laboratories, Columbus, Ohio.

** Supported in part by NSF MCS-79-12688.

The STILE Virtual Machine

The underlying "virtual machine" on which STILE graphs "run" is quite simple. A (STILE) graph is a directed graph in which the nodes are called boxes and the arcs are called links. A box is either a primitive box, which is provided by the system; or a simple box, which is created by the user by writing sequential code in a certain fashion outlined below; or a composite box, or subsystem, which is just shorthand notation for another STILE graph.

Each box has inputs and outputs. Each input may have at most one link coming into it, and each output may have at most one link going out of it. The computations are performed by boxes, which are activated under specific conditions described below. The links are used to transmit data from outputs of boxes to inputs of other boxes. Each link holds at most one item of data, called a token, at any time, and tokens may be produced by the box for which the link is an output link, and consumed by the box for which it is an input link.

A box (of any type) can be though of as forever executing the following code, expressed using a variant of Dijkstra′s guarded command notation [6]:

```
DO
        FC1: S1;
        FC2: S2;
              .
              .
              .
        FCn: Sn;
OD.
```

This notation means that each box can be thought of as having its own serial processor that performs its computation. This processor monitors the firing conditions FCi, which concern only the availability of data on the box′s input and output links (or possibly timing conditions in the most general real-time setting), and when one of them is true, it chooses (arbitrarily) to execute the action Si corresponding to one of the true firing conditions. It then returns to monitor the FCi again, proceeding as before, and never stopping. The only interactions among boxes are via data tokens passing over links; the data available within a box is local to that box.

The precise semantics of links, and how tokens are produced and consumed by the boxes during the course of execution of actions, are well defined [12], but are not presented here. Suffice it to say that there are system-provided procedures to produce and consume tokens, and that these are operationally described in very simple terms by their effects on the links.

Primitive and simple boxes directly obey the semantics presented informally above, and they are specified (by the system developer or user) by literally writing the firing conditions and corresponding actions. Composite boxes, or subsystems, are entire graphs that may be placed in larger graphs simply by naming them an connecting them as any other box. The semantics of a composite box are that the entire large graph does exactly what it would do if the subsystem were replaced by its corresponding graph, as in a "macro" expansion. A primitive or simple box may have arbitrary firing conditions and may perform arbitrary actions, and the composition rules that allow subsystems to be inserted into larger graphs preserve the semantics of primitive and simple boxes for composite boxes. Therefore, the user need not know whether a box is primitive, simple, or composite when constructing a graph, and in fact probably should not know in many cases. This means that subsystems can be reused without regard to their internal structure.

A Processor for Real-Time STILE Programs

Our real-time processor consists of computational units (CU′s), which may be microprocessors, or simpler or more complex processors, interconnected by a dedicated packet-switching network (as in Figure 1) based on the packet-switching chip set described in the next section. Each CU may be responsible for execution of a STILE box or perhaps a subgraph, depending on the CU′s capabilities. Tokens are passed between boxes in different CU′s via the communication network, with each token encapsulated in a packet that is routed to its destination by the packet-switchers (PS′s).

In our previous paper [1], we proposed a tree-based topology for such a machine. Further work has led us to conclude that a more general topology would be possible if certain extensions were made to the switching units we proposed there. As a result, a network constructed using the PS′s described here may take on arbitrary shapes depending on the shape of the STILE graph being executed. The only Constraints are relatively mild degree restrictions on the various CU′s and PS′s. For the purposes of this paper, we will assume that each CU has degree one and each PS has degree at most four in the physical interconnection network. In fact, these limits will depend on the pin limitations of the chips involved.

The CU′s of the real-time processor represent operators specifically designed for the real-time environment. This real-time constraint requires that the mapping of the programs onto the architecture must attempt to minimize communication costs. Fortunately it also implies that the processes running in this architecture are nonending and static. For this reason we can perform optimizations on the architecture and the mapping onto the architecture to insure that boxes that must communicate with each other can be placed in computational units that are physically close to each other (requiring a minimum number of jumps in the network).

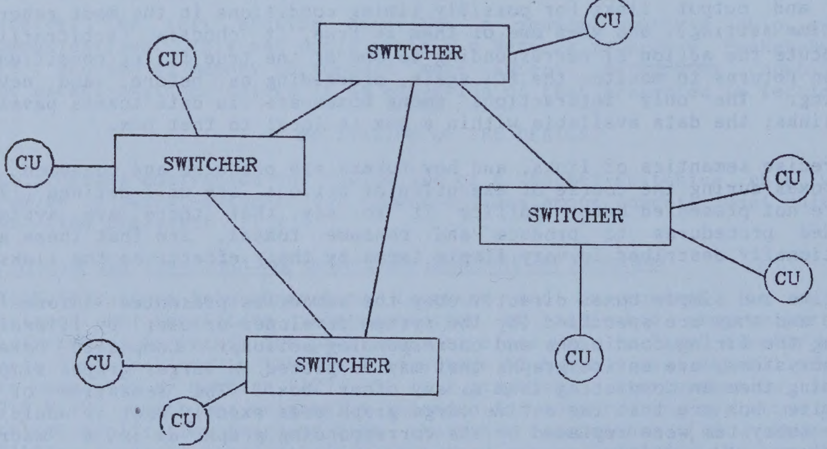


Figure 1. Sample Network of the Real-time Processor

A PACKET-SWITCHING CHIP FOR THE REAL-TIME PROCESSOR

The primary concern is to produce a packet-switcher that will work efficiently in the previously mentioned processor. This chip will treat all messages uniformly. Therefore, the chip will not be concerned with whether it is connected to a computational unit or to another communications chip. The rest of this section will describe the functions of the chip and an architectural overview.

Input/Output - Serial and Asynchronous

Because of the number of computational units in a system and the limitation on the number of pins available on a chip, all of the Input/Output to the packet-switcher will be bit serial. This should not be a great concern, even under the real-time constraint, because the tokens that will be passed through the chip will generally be small (i.e., a header followed by an integer value). Additionally, the input to the chip will be asynchronous since the computational units operate based upon the availability of data, and are not constrained to operation on a common clock. Later, we will describe how to increase bandwidth by configuring the packet-switchers in parallel in bit-slice fashion.

Routing - Strip and Send

Because of the real-time constraints of the system, a simplified design can be pursued. In this design, routing information is contained in the header information portion of the message rather than in the packet-switcher. This routing information consists of a list of output ports to be used. As each message reaches a packet-switcher, the first two bits (remember that the input/output degree of each packet-switcher is four) are removed from the message and used to route the message to an output port of the packet-switcher. Therefore, the messages shrink as they pass through the network.

This design is feasible because the real-time program is static, and once started, never ends. At compile time the routing information can be compiled into the programs. The seemingly large overhead of carrying all of the routing information in each packet may, at first, appear to violate the real-time constraints. However, one must keep in mind the optimization techniques that will be used will insure few jumps and therefore little overhead. When compared to standard routing techniques on a network of a large number of nodes, one may note the following advantages:

1. Additional memory is not required for routing tables in the packet-switcher.

2. Time is not lost during a table look-up for the next output port since the routing information is available in the message.

3. Provided that an effective mapping can be obtained, there is very little overhead in the header.

Functional Overview

Functionally the packet-switching chip can be viewed as a set of producer-consumer processes, each having three producers and one consumer. Each input port (except for the one associated with the destination output port) acts as a producer, and the destination output port represents the consumer. This will require on-board memory to buffer messages against delays and backup. *

POSSIBLE FURTHER USES AND CAPABILITIES

With a few extensions to the previously described chip a more general packet-switcher can be built with some of the capabilities of existing protocols and chips. What capabilities are not added to the extended version may be picked up from the already exiting chips.

---

* This design is being implemented in a software communication controller for a network of PDP-11/23's at OSU.

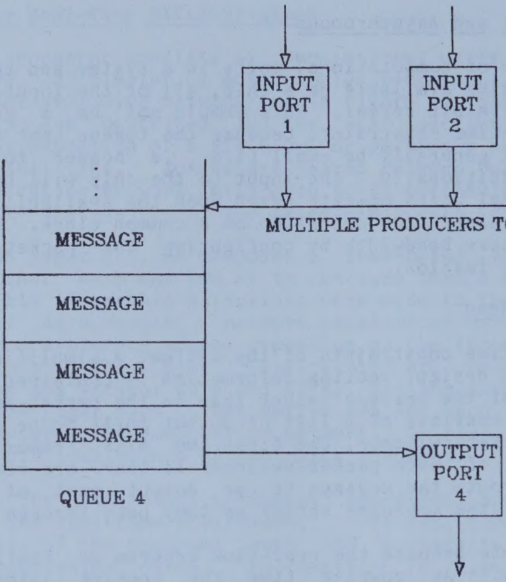


Figure 2. Producer-consumer example

Background Information

The following represents a list of various protocol support hardware [5,7,11].

a. Parity Generators and Checkers
b. Cyclic Redundancy Check Generators
c. Universal Asynchronous Receiver Transmitter - UART
d. Universal Synchronous Receiver Transmitter - USRT
e. Universal Synchronous Asynchronous Receiver Transmitter - USART
f. Data Encryption Units
g. Protocol Oriented Chips for HDLC and SDLC
h. Microprocessor Chips

In addition we find communications controllers like the Digital Equipment Corporation DMC11 synchronous line interface that implements the DEC protocol Digital Data Communications Message Protocol (DDCMP) through the use of an on board microprocessor.

A PACKET-SWITCHING CHIP SET - AN EXTENSION

Because of flexibilities in the design and the growing number of gates that can be placed on a chip through VLSI, a logical step would be to incorporate some of the capabilities mentioned previously into the design of the packet-switching chip. This section will describe the extensions we view as particularly valuable.

A Low Level Packet Switching Chip

The heart of the communications chip set is still a low level packet switching chip similar to the initial design. Several routing options and modes of operation are considered for this new chip.

Bit Maps - One of the more novel routing methods performed today uses a bit map for an address. In this method each bit of the address field represents a node in the network. While this method has the unfortunate quality of potentially large address fields, it also has the capability of broadcast messages to subsets of the nodes in the network. This method requires routing tables in the packet-switcher. The table would be loaded with information telling which output ports should receive the packet. Additionally, information would have to be loaded into the packet-switcher to indicate the length of the address portion, or the packet would have to carry this information in the header. This would also cause the functions carried out in the chip to be more complicated, since packets could be routed to several output ports.

Strip and Send - The strip and send operation would be identical to the previous description of the routing in the real-time processor.

Standard Table Look Up - A standard table look up is the method in most common use today. In this method the address field in the header portion of each packet is used to address a routing table to determine the output port to route the packet on.

Gateways - The purpose of a gateway is to allowing extended routing. When the number of bits in the link level protocol is insufficient to represent all of the nodes in the network, this method will allow for reuse of the address field by placing two sets of level two overhead on the message. The outside layer is used to transport the message to a gateway, where it is stripped off, revealing the layer that will be used to route on the other side of the gateway.

A gateway has two implications, depending on the routing algorithm in use. With the standard table-look-up algorithm, all the link level information is stripped off the packet to reveal another set of link level overhead. At this point the normal routing functions will be carried out. For the strip and send algorithm and the bit-map algorithm there is no special meaning, because additional bits or output port fields are added as the network expands.

An Add-On Memory Chip for Extended Routing

Due to the addition of extra functions and the requirement of handling larger messages than the tokens of the real-time processor, the memory to buffer messages and contain routing tables may not be large enough if restricted to being on the packet switching chip. To alleviate this problem, an additional chip may be added to provide the additional memory.

A Network Interface Chip

The purpose of this chip is to provide an interface between the nodes of the network and the packet switchers.

RS232-C - The purpose of this option is to allow the host to believe that it is communicating to a standard RS232-C interface.

Standard - The standard option allows the capability to input data in various formats (1 to n bits in parallel), and output that data in the same manner. Each of the outputs will be connected to one of the input ports of a packet switching chip. It will be the responsibility of this chip to duplicate the header information of the messages input to it on each of the output ports. This allows the user to view the network as n bit parallel, and also to have the capability of increasing throughput by transmitting information in parallel by connecting the packet-switching chips in bit-slice fashion (see Figure 3).

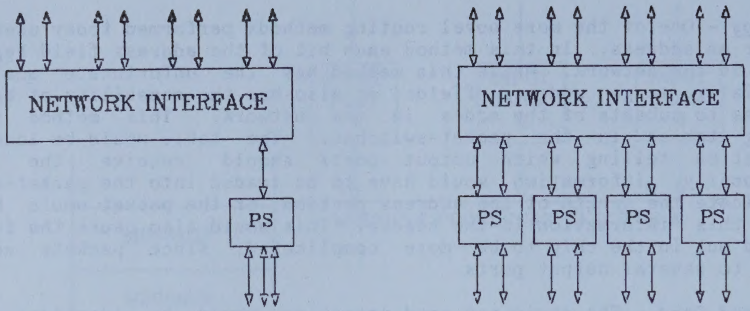


Figure 3. Examples of Network Interface and Packet-Switcher combinations.

SUMMARY

This paper has introduced the requirements and logical design of a general-purpose packet-switching chip set. The primary chip was designed to perform a packet-switching function in a real-time processor. Its general design allowed an extension to a more general-purpose chip set to perform packet switching.

ACKNOWLEDGEMENTS - We would like to acknowledge the contributions of the other members of our research group: José Alegría, Nena Bauman, Fred Glover, and I-pieng Kao.

BIBLIOGRAPHY

[1] Brown, M. E., and Weide, B. W. "Preliminary Design of a Highly Parallel Architecture for Real-Time Applications," Proceedings 18th Annual Allerton Conference on Communication, Control, and Computing, (October, 1980), pp. 534-543.

[2] Dennis, J. B. "First Version of a Data-Flow Procedure Language," Lecture Notes in Computer Science 19. Springer-Verlag, Heidelberg, 1974, pp. 362-376.

[3] Dennis, J. B. "Data Flow Supercomputers," Computer 13, 11 (November 1980), pp. 48-56.

[4] Dennis, J. B., and Misunas, D. P. "A Preliminary Architecture for a Basic Data-Flow Processor," SIGARCH News 3 (Proceedings 2nd Annual Symposium on Computer Architecture), 4 (Dec. 1974), pp. 126-132.

[5] Digital Equipment Corporation. Terminals and Communications Handbook. Maynard, Massachusetts, 1979.

[6] Dijkstra, E. W. A Discipline of Programming. Prentice-Hall, Englewood Cliffs, N. J., 1976.

[7] Intel Corporation. MCS-86 * Product Description. Santa Clara, California, February, 1979.

[8] MacQueen, D. B. Models for Distributed Computing. IRIA report no. 351, April 1979.

---

* MCS-86 is a registered trademark of the Intel Corporation.

[9] Milner, R. *Flow graphs and Flow Algebras*. Dept. of Computer Science, Univ. of Edinburgh, CSR-5-7, July 1977.

[10] Milner, R. *Algebras for Communicating Systems*. Dept. of Computer Science, Univ. of Edinburgh, CSR-25-78, June 1978.

[11] Texas Instruments Incorporated. *The TTL Data Book for Design Engineers*. Dallas, Texas, 1976.

[12] Weide, B. W., and Brown, M. E. "STILE: A STructure Interconnection Language and Environment," in preparation.

APPLICATION OF 2-D BIN PACKING ALGORITHMS FOR TASK SCHEDULING IN PASM

DAVID LEE TUOMENOKSA and HOWARD JAY SIEGEL
School of Electrical Engineering, Purdue University
West Lafayette, Indiana 47907

Two-dimensional bin packing algorithms pack a finite list of rectangles into a semi-infinite rectangular bin so as to minimize the height used. These algorithms have been applied to the scheduling of tasks with a shared resource (such as assigning computer memory to jobs) and to cutting stock (such as wood). Two-dimensional bin packing algorithms may also be applied to scheduling processing elements in multimicrocomputer systems such as PASM [2]. PASM is a partitionable SIMD/ MIMD multimicrocomputer system which requires multiprocessor tasks to be confined to a power of two processing elements. By applying this power of two confinement, the minimum value of the absolute worst case bound for the first-fit shelf algorithm [1] may be decreased by over 15 % as compared to the general case.

Shelf algorithms are 2-D bin packing algorithms which pack the rectangles onto shelves in the order given without sorting [1]. The rectangles are packed onto shelves each of height r^k for $0 < r < 1$ and some integer k. A rectangle of height h must be packed onto a shelf of height r^k where $r^{k+1} < h \leq r^k$. Therefore each rectangle is at least r times as high as the shelf onto which it is packed. Thus, r determines the amount of waste possible above a given rectangle on a shelf. The first-fit shelf (FFS) packing algorithm packs each rectangle as far *left* as possible onto the *lowest* shelf which is of the appropriate height. If the appropriate shelf does not exist, a shelf is created and the rectangle is placed onto it.

In [1] an absolute worst case bound is derived for the performance of the FFS algorithm which has a minimum value (at r =0.62) of 7.0 times the optimal packing. When the case of packing multiprocessor tasks which require a power of two processing elements is considered, a new absolute worst case bound is derived. The problem is first restricted to tasks of equal execution time and solved using the first-fit 1-D bin packing algorithm. This is then expanded to the 2-D case allowing tasks of different execution times. The power of two confinement reduces the minimum value (at r=0.58) of the absolute worst case bound to 5.8 times the optimal packing, yielding a 15 % decrease. This result will apply to any system with the "power of two" size confinement.

An implementation of this algorithm is being developed for PASM. The FFS task scheduler will run on the System Control Unit as part of the PASM task manager. One process determines on which shelf an incoming task should be placed. A second process determines which shelf's tasks are next to be loaded into memory and run by the Parallel Computation Unit (PCU), which contains N processors, N memories, and an interconnection network. Aspects of PASM which effect the scheduler include: the double buffered PCU memories, the power of two partitioning, and the rules for which PCU processors may be joined to form a partition.

The above material will appear in detail in a Purdue technical report to be published.

[1] B. S. Baker and J. S. Schwarz, "Shelf algorithms for two-dimensional packing problems," *Conf. on Inf. Sci. and Sys.*, Mar. 1979, pp. 273-276.

[2] H. J. Siegel, L. J. Siegel, F. C. Kemmerer, P. T. Mueller, Jr., H. E. Smalley, Jr., and S. D. Smith, "PASM: a partitionable SIMD/MIMD system for image processing and pattern recognition," *IEEE Trans. Computers*, Scheduled to appear Dec. 1981.

This research was supported by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under grant number AFOSR-78-3581, and by a Purdue University Graduate Fellowship. The United States Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation hereon.

EXPECTED CAPACITY OF BINOMIAL CONCENTRATORS

GERALD M. MASSON[1]
DEPARTMENT OF ELECTRICAL ENGINEERING
THE JOHNS HOPKINS UNIVERSITY
BALTIMORE, MARYLAND 21218

AND

S. BRENT MORRIS
DEPARTMENT OF DEFENSE
FORT GEORGE G. MEADE, MARYLAND 20755

ABSTRACT

A concentrator is an interconnection network with n inputs and m outputs, $n > m$, wherein a specified subset of inputs can be simultaneously connected to some equal-sized subset of outputs. The primary measure used for the performances of concentrators is an upper bound on the number of specified inputs which can always simultaneously be connected to outputs. This guaranteed throughput is called actual capacity.

In many applications, however, a more practical measure of a concentrator's capability is a probabilistic measure of its throughput in the following sense: given an input subset of size k, $k \leq m$, what is the average number of inputs that can be connected to outputs? This measure will be called expected capacity. This paper considers the expected capacity of a class of sparse crossbar designs called binomial networks.

INTRODUCTION

A concentrator is an interconnection network with n inputs and m outputs, $n > m$, wherein a specified subset of inputs can be simultaneously connected to some equal-sized subset of outputs. For concentrators, unlike permuters, the inputs can always be specified, but the outputs cannot. In spite of this limitation, concentrators are of significant importance to a wide range of connection applications [3].

The primary measure used for the performance of a concentrator is an upper bound on the number of inputs which can always simultaneously be connected to outputs. This guaranteed throughput is called actual capacity and is denoted as c. One of the main issues in the study of concentrators is determining the minimum number of crosspoints, or switching elements, required to achieve some actual capacity for a given number of inputs and outputs [1]-[7].

In many applications, however, a more practical evaluation of a concentrator's capability is a probabilistic measure of its throughput in the following sense: given an input subset of size k, $k \leq m$, what is the average number of those k inputs that can simultaneously be connected to outputs? This measure will be called expected capacity and will be denoted as $e(k)$.

To reinforce the significance of the expected capacity measure and to underscore the possibly misleading nature of the actual capacity measure, consider the 6-input to 4-output sparse crossbar concentrator shown in

---

[1] G. M. Masson's involvement in this research was sponsored in part of Office of Naval Research Contract N00014-75-0-1196 and the Ballistic Missle Defense Agency Contract DAAG29-76-D-0100.

Figure 1. Since inputs 5 and 6 cannot simultaneously be connected to outputs, it follows that $c = 1$. Yet if we consider all 15 choices of 4 inputs, we see that 9 are such that all 4 can simultaneously be connected to the outputs and 6 are such that only 3 can be simultaneously connected to outputs. Hence,

$$e(4) = \frac{9 \cdot 4 + 6 \cdot 3}{15} = 3.6 .$$

Furthermore, $e(3) = 2.8$ and $e(2) = 1.93$. The somewhat pessimistic actual capacity measure of $c = 1$ is thus seen to be countered by the expected capacity measure. It is our contention that these expected capacity measures are more useful relative to the evaluation of the potential performance of concentrators in applications.

The expected capacity of a class of sparse crossbar concentrator designs called binomial networks [6] will be considered in this paper. It will be seen that in spite of the rather marked sparseness of crosspoints, the expected capacity will be impressively high.

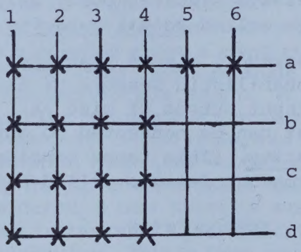


Figure 1

Preliminaries

Any sparse crossbar network can be described as a triplet (I, O, R), where I is the input set, $|I| = n$; O is the output set, $|O| = m$; and R is a relation between I and O. For $i \in I$ and $o \in O$, $o \in R(i)$ implies there exists a crosspoint between input i and output o. R specifies the design of the network. For any $I \subseteq I$, $R(I) = \bigcup_{i \in I} R(i)$. R can graphically be expressed as a crosspoint diagram such as that shown in Figure 2. In this figure, $I = \{1,2,\ldots,28\}$, $O = \{a,b,\ldots,g,h\}$, and $R(1) = \{a,b\}$, $R(2) = \{a,c\},\ldots,R(28) = \{g,h\}$. An "×" on the grid between input i and output o implies that $o \in R(i)$.

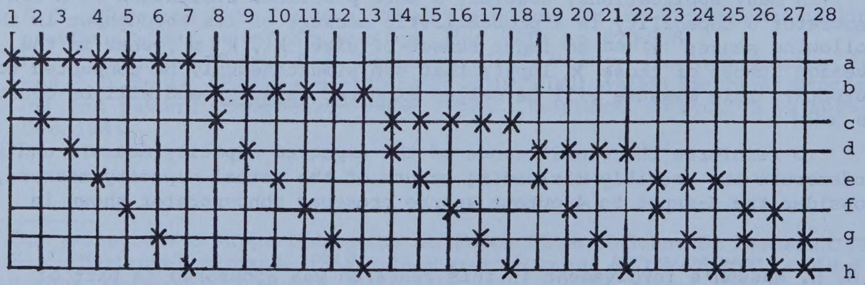


Figure 2

The particular sparse crossbar network that we will be concerned with here is the binomial network first suggested by Masson [3]. For integers m and v , v < m , in an $\binom{m}{v}$-network: $\mathcal{I} = \{1,2,\ldots,\binom{m}{v}\}$, $|\mathcal{O}| = m$, $|\mathcal{R}(i)| = v$, and $\mathcal{R}(i) \neq \mathcal{R}(j)$ for all $i \neq j$. Figure 2 is an $\binom{8}{2}$-network. When used as a concentrator design, an $\binom{m}{v}$-network has an actual capacity of $c = v+2$ [3]. Recently, Nakamura and Masson [7] have further shown that such concentrator designs contain a minimum number of crosspoints and, for some choices of m and v , are unique, minimum designs.

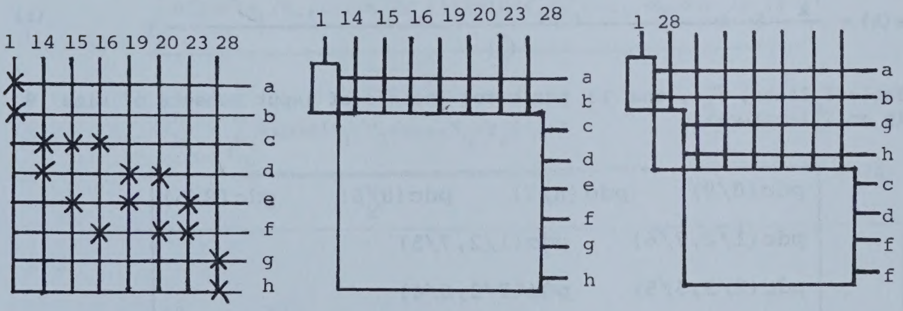
Decompositions of Input Sets

Consider any $I \subset \mathcal{I}$, $|I| = k \leq m$. A decomposition of I is an association of the subsets of a partitioning of I , denoted as $(I_1,\ldots,I_q)$, with the subsets of an equal-sized partitioning of an $O \subset \mathcal{O}$, denoted as $(O_1,\ldots,O_q)$, where for all $1 \leq i \leq q$, $I_i \neq \phi \neq O_i$, and

$$\mathcal{R}(I_i) = O_i .$$

This decomposition is denoted as $(I_1/O_1,\ldots,I_q/O_q)$ and each I_i/O_i is called a component of the decomposition. A component of a decomposition is a prime component if it cannot be further decomposed. A prime decomposition contains only prime components. It should be clear that for any $I \subset \mathcal{I}$, $|I| = k \leq m$, there are in general many decompositions but only one prime decomposition.

Example 1: Consider an input subset of the $\binom{8}{2}$-network of I = {1,14,15, 16,19,20,23,28} as shown in Figure 3(a). A decomposition of I is ({1},{a,b} , {14,15,16,19,20,23,28}/{c,d,e,f,g,h}) . This is indicated in Figure 3(b). This decomposition is not prime since the component {14,15, 16,19,20,23,28}/{c,d,e,f,g,h} can be further decomposed. The prime decomposition of I is ({1}/{a,b} , {28}/{g,h} , {14,15,16,19,20,23}/{c,d,e,f}). This is shown in Figure 3(c).



(a) (b) (c)

Figure 3

Example 2

Consider I = {1,14,15,16,17,18,23,28} . In this case, the prime decomposition is ({1/{a,b} , {14,15,16,17,18,23,28}/{c,d,e,f,g,h}) .

We can form classes of decompositions based on the sizes of the input and output subsets. In general, if a decomposition of I is $(I_1/O_1,\ldots, I_q/O_q)$, we will say that I is a member of decomposition class $(|I_1|/|O_1|, \ldots, |I_1|/|O_1|)$, written $dc(|I_1|/|O_1|,\ldots,|I_q|/|O_q|)$. Similarly, if this decomposition of I is prime, we will say that I is a member of pdc($|I_1|/|O_1|,\ldots,|I_q|/|O_q|$). We can equivalently say that (p)dc($|I_1|/|O_1|$, $\ldots,|I_q|/|O_q|$) covers I. Each $|I_i|/|O_i|$ is called a factor of the (p)dc. In Example 2, I is a member of pdc(1/2,7/6). Some dc's are inherently prime in that no member of the dc can ever be further decomposed. For example, dc(8/5) and dc(1/2,1/2,6/4) for choices of 8 inputs in the $\binom{8}{2}$-network are inherently prime. However, as we saw in Example 1, dc(1/2,7/6) is not inherently prime; indeed, each input subset of size 8 covered by dc(1/2,7/6) is either a member of pdc(1/2,7/6), pdc(1/2,7/5), or pdc(1/2,1/2,6/4).

Expected Capacity

When considering concentration throughput, it should be clear that for a given subset of inputs, the number of inputs which can simultaneously be connected to outputs is dependent upon the prime decomposition of that set of inputs. In Example 1, we saw that for the $\binom{8}{2}$-network, I = {1,14,15,16, 19,20,23,28} was a member of pdc(1/2,1/2,6/4). As is clear from Figure 3(c), it follows that only 6 of the 8 inputs of I can simultaneously be connected to outputs: namely, input 1 can be connected to outputs a or b; input 28 can be connected to outputs g or h, and any four of the remaining six inputs, 14,15,16,19,20, and 23, can be connected to outputs c,d,e, and f. Similarly, for Example 2 where I = {1,14,15,16,17,18,23,28}, only 7 of the 8 inputs can simultaneously be connected to outputs. In general, suppose I is a member of pdc($x_1/y_1,x_2/y_2,\ldots,x_q/y_q$). Then the number of inputs in I that can simultaneously be connected to outputs is $\min(x_1,y_1)+\min(x_2,y_2)+\ldots+\min(x_q,y_q)$.

Given all $\binom{n}{k}$ choices of input subsets of size k, we will let N[pdc($x_1/y_1,\ldots,x_q/y_q$)], $x_1+\ldots+x_q = k$, denote the total number of such choices which are members of pdc($x_1/y_1,\ldots,x_q/y_q$). Let $\mathcal{P}_k$ denote the set of all pdc's for k inputs. Then the expected capacity, e(k), is

$$e(k) = \frac{\sum\limits_{\mathcal{P}_k} N[\mathrm{pdc}(x_1/y_1,\ldots,x_q/y_q)]\cdot(\min(x_1,y_1)+\ldots+\min(x_q/y_q))}{\binom{n}{k}} \tag{1}$$

Table 1 lists, $\mathcal{P}_8$, the 13 pdc's for choices of input subsets of size 8 in an $\binom{8}{2}$-network.

pdc(8/8)	pdc(8/7)	pdc(8/6)	pdc(8/5)
pdc(1/2,7/6)	pdc(1/2,7/5)		
pdc(2/3,6/5)	pdc(2/3,6/4)		
pdc(3/3,5/5)	pdc(3/3,5/4)		
pdc(3/4,5/4)			
pdc(4/4,4/4)			
pdc(1/2,1/2,6/4)			

Table 1

Failing Prime Decomposition Classes

To compute $e(k)$, we must first generate the set $\mathcal{P}_k$. It is sufficient to determine all pdc's in which there is at least one factor, say, x/y, where $x > y$. These will be referred to as failing pdc's. The term "failing" comes from the fact that the binomial network cannot simultaneously connect all k inputs of a failing pdc to k outputs. Examples 1 and 2 both gave input subsets that were members of failing pdc's. Let $\mathcal{F}_k$, denote the set of all failing pdc's in $\mathcal{P}_k$. It is clear that when we know $\mathcal{F}_k$, all the other pdc's in $\mathcal{P}_k$ must be such that its members consist only of subsets of k inputs which all can indeed be simultaneously connected to outputs. From Table 1 for the $\binom{8}{2}$-network and $k = 8$, $\mathcal{F}_8 = \{\mathrm{pdc}(8/7)$, $\mathrm{pdc}(8/6)$, $\mathrm{pdc}(8/5)$, $\mathrm{pdc}(1/2,7/6)$, $\mathrm{pdc}(1/2,7/5)$, $\mathrm{pdc}(2/3,6/5)$, $\mathrm{pdc}(2/3,6/4)$, $\mathrm{pdc}(3/3,5/4)$, $\mathrm{pdc}(3/4,5/4)$, $\mathrm{pdc}(1/2,1/2,6/4)\}$.

Generating $\mathcal{F}_k$ for an $\binom{m}{v}$-network is straightforward. We begin by first determining the set of possible factors for all pdc's of the $\binom{m}{v}$-network. It is not difficult to see that in an $\binom{m}{v}$-network, x/y can be a factor of a pdc if $v \le y \le m$ and $\left\lceil \frac{y-v}{v-1} \right\rceil + 1 \le x \le \binom{y}{v}$. Whether or not some x/y satisfying these conditions is a member of a pdc depends on k.

Table 2 shows all the possible factors of pdc's for the $\binom{8}{2}$-network. Since for subsets of inputs of size k, any pdc of an $\binom{m}{v}$-network contains a selection of these factors such that $x_1+\ldots+x_q = k$ and $y_1+\ldots+y_q \le m$, it is straightforward to generate Table 1 from Table 2. It is even simpler to generate $\mathcal{F}_k$ from Table 2 as we can simply augment the selection process with the condition that for at least one factor in each of the pdc's we must have that $x > y$.

Enumeration of $N[\mathrm{pdc}(x_1/y_1,\ldots,x_q/y_q)]$

Given $\mathcal{F}_k$, we can determine the expected capacity of an $\binom{m}{v}$-network as

$$e(k) = \frac{\sum_{\mathcal{F}_k} N[\mathrm{pdc}(x_1/y_1,\ldots,x_q/y_q)]\cdot(\min(x_1/y_1)+\ldots+\min(x_q/y_q))}{\binom{n}{k}} + \frac{\left(\binom{n}{k} - \sum_{\mathcal{F}_k} N[\mathrm{pdc}(x_1/y_1,\ldots,x_q/y_q)]\right)\cdot k}{\binom{n}{k}} \tag{2}$$

1/2							
	2/3	3/3					
		3/4	4/4	5/4	6/4		
			4/5	5/5	6/5	7/5	8/5
				5/6	6/6	7/6	8/6
					6/7	7/7	8/7
						7/8	8/8

Table 2

For the $\binom{8}{2}$-network and input subset choices of size 8, the enumeration of the members in each pdc in $\mathcal{F}_8$ will serve to illustrate a straightforward general procedure. In particular, at various points we will indicate how the calculation of the number of members in each pdc in $\mathcal{F}_8$ would also pertain to $\binom{m}{2}$-networks as well (assuming, of course, that $m \geq 8$). These calculations will then be placed in what we will refer to as a Prime Component Count Table (see Table 3) so that for other evaluations of (2) for $\binom{m}{2}$-networks, the enumerations of the following type need not be repeated. Indeed, the availability of such a table so simplifies the evaluation of (2), that an Expected Capacity Table (see Table 3) can be easily generated for $\binom{m}{2}$-networks.

To begin, note that there are $\binom{28}{8} = 3,108,105$ input subsets of size 8. We will start our enumeration by considering N[pdc(8/7)]. Suppose we are given some specific choice of 7 of the 8 outputs. The following observations relative to these 7 outputs are fundamental to enumerating N[pdc(8/7)]:

(1) The number of subsets of 8 inputs with crosspoints only to these specific 7 outputs is $\binom{\binom{7}{2}}{8} = \binom{21}{8} = 67,830$.

(2) Of these 67,830 subsets, each is covered by pdc(8/7), pdc(8/6), pdc(8/5), pdc(1/2,7/5), pdc(2/3,6/4), or pdc(3/3,5/4).

(3) The number of subsets of 8 inputs with crosspoints to only 6 or 5 of these 7 outputs is, by the Principal of Inclusion and Exclusion,

$$\binom{7}{6}\binom{\binom{6}{2}}{8} - \binom{7}{5}\binom{\binom{5}{2}}{8}.$$

This accounts for those subsets covered by pdc(8/6) and pdc(8/5).

(4) The number of subsets of 8 inputs covered by pdc(1/2,7/5) relative to these 7 outputs is

$$\binom{7}{2}\binom{\binom{2}{2}}{1}\binom{5}{5}\binom{\binom{5}{2}}{7}.$$

(5) The number of subsets of 8 inputs covered by pdc(2/3/6/4) relative to these 7 outputs is

$$\binom{7}{3}\binom{\binom{3}{2}}{2}\binom{4}{4}\binom{\binom{4}{2}}{6}.$$

(6) The number of subsets of 8 inputs covered by pdc(3/3,5/4) relative to these 7 outputs is

$$\binom{7}{3}\binom{\binom{3}{2}}{3}\binom{4}{4}\binom{\binom{4}{2}}{5}.$$

(7) Given the 8 outputs in the $\binom{8}{2}$-network, there are $\binom{8}{7}$ choices of 7 of these 8 outputs.

Now, from Observations (1) to (6), given 7 specific outputs, the number of choices of 8 inputs covered by pdc(8/7) is

$$\begin{aligned}&\binom{21}{8} - \left(\binom{7}{6}\binom{15}{8} - \binom{7}{5}\binom{10}{8}\right)\\&- \binom{7}{2}\binom{10}{7} - \binom{7}{3}\binom{3}{2} - \binom{7}{3}\binom{6}{5}\\&= 156,555. \qquad (3)\end{aligned}$$

With (3) and Observation (7), in the $\binom{8}{2}$-network, it follows that

$$\begin{aligned}N[\text{pdc}(8/7)] &= \binom{8}{7}(156,555)\\&= 1,252,440. \qquad (4)\end{aligned}$$

Before going on to other enumerations for this $\binom{8}{2}$-network, note that if we were more generally considering an $\binom{m}{2}$-network where $m \geq 8$, and were interested in N[pdc(8/7)] for this $\binom{m}{2}$-network, it would still be necessary to calculate (3), and then determine

$$N[pdc(8/7)] = \binom{m}{7}(156{,}555) \, .$$

Hence, (3) is a general result pertaining to the number of choices of 8 inputs covered by the factor (8/7) in a pdc in any $\binom{m}{2}$-network (where, of course, $m \geq 8$). We will call (3) a Prime Component Count and denote it as

$$PCC(8/7) = 156{,}555$$

for $\binom{m}{2}$-networks. This will later be used as an entry in the Prime Component Count Table (PCC-Table) for $\binom{m}{2}$-networks.

Returning now to our example of the $\binom{8}{2}$-network, it should be clear that by Observations (4) and (7),

$$\begin{aligned} N[pdc(1/2,7/5)] &= \binom{8}{7}\binom{7}{2}\binom{10}{7} \\ &= 20160 \, , \end{aligned} \qquad (5)$$

which can also be written as

$$N[pdc(1/2,7/5)] = \binom{8}{2}(1)\binom{6}{5}(120) \, .$$

Similarly, by Observations (5) and (7)

$$\begin{aligned} N[pdc(2/3,6/4)] &= \binom{8}{7}\binom{7}{3}\binom{3}{2} \\ &= \binom{8}{3}(3)\binom{5}{4}(1) \\ &= 840 \, , \end{aligned} \qquad (6)$$

and by Observations (6) and (7)

$$\begin{aligned} N[pdc(3/3,5/4)] &= \binom{6}{7}\binom{7}{3}\binom{6}{5} \\ &= \binom{8}{3}(1)\binom{5}{4}(6) \\ &= 1680 \end{aligned} \qquad (7)$$

Again, if we were more generally to consider these last three enumerations for the more general case of an $\binom{m}{2}$-network, $m \geq 8$, it is not difficult to see that

$$N[pdc(1/2,7/5)] = \binom{m}{2}(1)\binom{m-2}{5}(120) \, ,$$

$$N[pdc(2/3,6/4)] = \binom{m}{3}(3)\binom{m-3}{4}(1) \, ,$$

and

$$N[pdc(3/3,5/4)] = \binom{m}{3}(1)\binom{m-3}{4}(6) \, ,$$

where

$$\begin{aligned} PCC(1/2) &= 1 \\ PCC(7/5) &= 120 \\ PCC(2/3) &= 3 \\ PCC(6/4) &= 1 \\ PCC(3/3) &= 1 \end{aligned}$$

and

$$PCC(5/4) = 6$$

Finally, to calculate N[pdc(8/6)] and N(pdc(8/5)], we can proceed similarly to our consideration of N[pdc(8/7)]. That is, for N[pdc(8/6)], suppose we are given some specific choice of 6 of the 8 outputs. Observe that:

(8) The number of subsets of 8 inputs with crosspoints only to these specific 6 inputs is $(\binom{6}{2}) = 6435$.

(9) Of these 6435 subsets, each is covered by pdc(8/6) or pdc(8/5).

(10) The number of subsets of 8 inputs with crosspoints to only 5 of these 6 outputs is $\binom{6}{5}\binom{\binom{5}{2}}{8} = 270$.

(11) Given the 8 outputs in the $\binom{8}{2}$-network, there are $\binom{8}{6}$ choices of 6 of the 8 outputs.

Now, from Observations (8) to (10), given 6 specific outputs, the number of choices of 8 inputs covered by pdc(8/6) is

$$\binom{15}{8} - \binom{6}{5}\binom{10}{8} = 6{,}165 \qquad (8)$$

With (8) and Observation (11), in the $\binom{8}{2}$-network it follows that

$$\begin{aligned} N[pdc(8/6)] &= \binom{8}{6}(6{,}165) \\ &= 172{,}620 \qquad (9) \end{aligned}$$

Finally, for N[pdc(8/5)], suppose we are given some specific choice of 5 of the 8 outputs. Relative to these 5 outputs, the number of choices of 8 inputs covered by pdc(8/5) is

$$\binom{\binom{5}{2}}{8} = 45 \qquad (10)$$

It follows that in an $\binom{8}{2}$-network,

$$\begin{aligned} N[pdc(8/5)] &= \binom{8}{5}(45) \\ &= 2520 \qquad (11) \end{aligned}$$

Note as before that if we were considering an $\binom{m}{2}$-network, $m > 8$, and we were interested in N[pdc(8/6)] and N[pdc(8/5)], we would still first calculate (8) and (10), respectively. Hence,

$$PCC(8/6) = 6{,}165$$

and

$$PCC)8/5) = 45$$

What remains to be accounted for are the pdc's in F_8 which cover input selections of size 8 that collectively have crosspoints to all 8 outputs, but still fail because some subset of the 8 inputs fails. These are pdc(1/2,7/6), pdc(2/3,6/5), pdc(3/4,5/4), and Pdc(1/2,12/,6/4). Using reasoning similar to the above, it follows that

$$\begin{aligned} N[pdc(1/2,7/6)] &= \binom{8}{2}\binom{\binom{2}{2}}{1}\binom{6}{6}\Bigl(\binom{\binom{6}{2}}{7} \\ &\quad - \binom{6}{5}\binom{\binom{5}{2}}{7} - \binom{6}{2}\binom{\binom{2}{2}}{1}\binom{4}{4}\binom{\binom{4}{2}}{6}\Bigr) \\ &= \binom{8}{2}(1)\binom{6}{6}(5700) \\ &= 159{,}600 \ , \qquad (13) \end{aligned}$$

where

$$PCC(7/6) = 5{,}700$$

Similarly, for the $\binom{8}{2}$-network,

$$\begin{aligned} N[pdc(2/3,6/5)] &= \binom{8}{3}\binom{\binom{3}{2}}{2}\binom{5}{5}\Bigl(\binom{\binom{5}{2}}{6} - \binom{5}{4}\binom{\binom{4}{2}}{6}\Bigr) \\ &= \binom{8}{3}(3)\binom{5}{5}(205) \\ &= 34{,}440 \ , \qquad (14) \end{aligned}$$

where

$$PCC(6/5) = 205 \ ;$$

$$N[pdc(3/4,5/4)] = 6,720 \qquad (15)$$

where

$$PCC(3/4) = 16 ;$$

and

$$N[pdc(1/2,1/2,6/4)] = 210 \qquad (16)$$

Similar expressions for $\binom{m}{2}$-networks for (15) and (16) should be clear.

Now, using results (4) to (15) in (2), we get $e(8) = 7.405$.

PCC Table and Expected Capacity Chart

The calculations of the previous section together with further results are shown in Table 3. This table is called the $\binom{m}{2}$-PCC Table. A fuller $\binom{m}{2}$-PCC Table is available in [8]. Also available in [8] are the $\binom{m}{3}$ to $\binom{m}{5}$-PCC Tables.

Table 4 is the $\binom{m}{2}$-Expected Capacity Table. Again, fuller $\binom{m}{2}$ to $\binom{m}{5}$-Expected Capacity Tables are available in [8].

Table 3 and 4, their extensions, and the additional tables in [8] were generated by means of an APL program that can be obtained from the authors.

Conclusions

As stated in the introductory section, it is our contention that the expected capacity measure gives a more useful assessment of the concentration throughput of binomial sparse crossbar concentrators. Furthermore, as seen from Table 4, this throughput measure is rather encouraging even for $\binom{m}{2}$-networks, which are the most sparse of all binomial designs, having only 2 crosspoints on each input. The $\binom{m}{3}$ to $\binom{m}{5}$ Expected Capacity Tables are even more encouraging, as usually the expected capacity is quite close to k.

References

1. M. Pinsker, "On the Complexity of a Concentrator," Proc. 7th International Teletraffic Conference Stockholm, 1973, pp. 318/1-318/4.
2. N. Pippenger, "On the Complexity of Strictly Nonblocking Concentration Networks," IEEE Trans. Comm., Vol. COM-22, 1974, pp. 1890-1892.
3. G. M. Masson, "Binomial Switching Networks for Concentration and Distribution," IEEE Trans. Comm. Vol. COM-25, No.9, 1977, pp. 873-883.
4. S. Nakamura and G. M. Masson, "Capacity Calculation of Composite Concentrators," Proc. of the Seventeenth Annual Allerton Conference on Communication, Control, and Computing, 1979, pp. 1016-1025.
5. S. Nakamura and G. M. Masson, "Higher Order Composite Concentrators," Proc. 14th Ann. Conf. on Information Sciences and Systems, 1980.
6. G. M. Masson, G. C. Gingher, and S. Nakamura, "A Sampler of Circuit Switching Networks," Computer, Vol. 12, No. 6, 1979, pp. 32-48.
7. S. Nakamura and G. M. Masson, "On the Design of Concentrators," in review.
8. G. M. Masson and S. B. Morris, Expected Capacity of Binomial Concentrators," in review.

						x			
		[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
	[1]	0	0	0	0	0	0	0	0	0
	[2]	1	0	0	0	0	0	0	0	0
	[3]	0	3	1	0	0	0	0	0	0
y	[4]	0	0	16	15	6	1	0	0	0
	[5]	0	0	0	125	222	205	120	45	10
	[6]	0	0	0	0	1296	3660	5700	6165	4945
	[7]	0	0	0	0	0	16807	68295	156555	258125
	[8]	0	0	0	0	0	0	262144	1436568	4483360
	[9]	0	0	0	0	0	0	0	4782969	33779340

Table 3

$\binom{m}{2}$-PCC Table

Given a column heading, say x , and a row heading, say, y , the entry in the corresponding position in the table is PCC(x/y).

						m				
	[1]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	[2]	1.000	2.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	[3]	1.000	2.000	3.000	0.000	0.000	0.000	0.000	0.000	0.000
	[4]	1.000	2.000	3.000	4.000	0.000	0.000	0.000	0.000	0.000
	[5]	1.000	2.000	3.000	4.000	4.881	0.000	0.000	0.000	0.000
k	[6]	1.000	2.000	3.000	4.000	4.970	5.730	0.000	0.000	0.000
	[7]	1.000	2.000	3.000	4.000	4.990	5.908	6.569	0.000	0.000
	[8]	1.000	2.000	3.000	4.000	4.996	5.962	6.819	7.405	0.000
	[9]	1.000	2.000	3.000	4.000	4.998	5.983	6.916	7.709	8.239

Table 4

$\binom{m}{2}$-Expected Capacity Table

The columns correspond to m ; the rows correspond to k . The entry in the i^{th} row , j^{th} column is e(i) for a $\binom{j}{2}$-network.

SOME PROPERTIES OF THE BINARY N-CUBE AS A NETWORK INTERCONNECTION STRUCTURE*

J. G. KUHL and S. M. REDDY
Department of Electrical and Computer Engineering
University of Iowa, Iowa City, Iowa 52242

ABSTRACT

The use of the binary n-cube as a model for the interconnection structure of large computing networks is discussed. Two distinct n-cube based interconnection structures which have been presented in the literature are considered. For each, results concerning network connectivity and distances between processors in the presence of failures are given. In addition, the ease of message routing is shown to be essentially the same for both types of networks. Finally, the results lead to the introduction of a new type of interconnection structure based on the edge-graph of the n-cube.

I. INTRODUCTION

The binary n-cube has received some attention as a potential model for the interconnection structure of large computing networks [1,2]. A binary n-cube is an undirected graph consisting of 2^n vertices. Each vertex is given a distinct n-bit binary "address", and an edge connects each pair of vertices whose addresses differ in exactly one bit.

Two distinct methods of employing the n-cube as a network interconnection structure have been proposed. The first of these [1] views the vertices of the n-cube as processing elements, with distinct interprocessor links represented by the edges of the n-cube. In such a system, interprocessor communication would be accomplished via packet or message switching. The desirability of using this n-cube based interconnection structure stems, among other things, from two well-known facts:

1) The diameter [3] (i.e. the maximum of the shortest distances between all pairs of vertices) of the n-cube is only n. Thus packets can reach their destinations over very short paths, making for efficient routing.

2) Routing of packets can be accomplished in a very simple, distributed fashion [1]. A packet, when constructed by its originator, can include an n-bit routing field which is the bit-by-bit exclusive - or of the address of the originator with the address of the destination. Then each node forwarding the packet (including the originator) simply picks a position i, $1 \leq i \leq n$ for which the routing tag is 1, sets this position of the routing tag to 0 and passes the message on to the neighboring node whose address differs from its own address in bit i. The packet will reach its destination when its routing tag becomes all zero, after having traversed a path of length h, where h is the hamming distance between the source and destination addresses. In this paper the type of n-cube based networks described above will be called Type I n-cube networks.

An alternative network structure, based on the n-cube, was given in [2]. In this work, the edges of the n-cube were viewed as representing processors, while each vertex of the n-cube was considered to represent a bus, shared by all edges (processors) incident on the vertex. This type of network, which will be referred to as a Type II n-cube network, would consist of $n\ 2^{n-1}$ processors, each connected two two buses, with each bus

---

* This work was supported, in part, under Air Force Office of Scientific Research Grant No. AFOSR-78-3582.

being shared by n processors. The total number of buses would be 2^n.

It can be easily shown [2] that, for Type II n-cube networks, the distance (i.e. number of buses a message must traverse) between processors is at most n. Furthermore, as will be demonstrated in a later section of this paper, routing of messages in a Type II n-cube network is nearly as easy as it is for a Type I network. Since this method of network configuration requires only two buses per processor, regardless of n, as opposed to the need for n I/O ports per processor for Type networks, it may be more amenable to modular expansion, and might be more cost effective for large n.

There are a number of factors, in addition to distances between processors, which can affect the suitability of a network interconnection strategy. One of the most commonly studied of these factors is the resiliency of the network to failures among its constituent processors, i.e. the ability of the network to maintain routes between remaining nodes of the network, when some nodes have been lost. Another consideration which may be important, but which has generally not been studied, is the question of how the loss of processors affects the distances between remaining processors in the system. This may be of concern because connectivity alone may be a meaningless measure if processor failures can cause distances between remaining processors to become so large that network efficiency is seriously impaired.

The next two sections of this paper will discuss the above mentioned factors for Type I and Type II n-cube networks. Section IV will present a routing algorithm for Type II n-cube networks and will discuss a new potential interconnect structure for Type I networks.

II. TYPE I N-CUBE NETWORKS

Since Type I n-cube networks directly represent processors as vertices and communication links as edges of the n-cube, questions concerning maximum distance between processors, resiliency of the network to processor failures, and the effect of processor failures on the distances between remaining processors, can all be answered by studying simple graph theoretic properties of the n-cube. We begin by formally defining two well-known concepts from graph theory.

Definition 1: Given an undirected graph G with set of edges E(G) and set of vertices V(G), the connectivity of G is the cardinality of the smallest subset of V(G) whose removal from G leaves two remaining vertices of G with no path between them.

Definition 2: The distance between two vertices, say a and b, of an undirected graph G, is the length of (i.e. the number of edges in) the shortest path connecting a and b in G.

Definition 3: The diameter of an undirected graph G, is the maximum of the distances between all pairs of vertices of G.

In addition, we make the following definition:

Definition 4: The r-deleted diameter of an undirected graph G, is the maximum of the diameters of all subgraphs of G obtained by removing r vertices from G. (This notion was first defined in [4], where it was called "diagnostic diameter").

It is well-known (e.g. [6]) that the connectivity of the n-cube is n. Hence a Type I n-cube network can tolerate up to n-1 processor failures, with the assurance that at least one route exists between any two remaining processors. The following theorems, which establish the r-deleted diameter $(0 \le r \le n)$ of the n-cube, resolve the question of the effect of failures on the distances between remaining processors.

Theorem 1: Let A and B be any two nodes of an n-cube whose addresses differ in d positions, $1 \le d \le n$. Then there are n vertex disjoint paths of length $\le d + 2$ connecting A and B, d of which are of length d and n-d of which are of length d + 2.

Proof: This can be shown constructively. The proof is omitted here due to its length. The complete proof can be found in [5].

Q.E.D.

Theorem 2: The r-deleted diameter ($0 \leq r < n-1$), of the n-cube is r and the (n-1)-deleted diameter is n+1.

Proof: Follows directly from Theorem 1.

Q.E.D.

The result of Theorem 2 indicates that Type I n-cube networks are quite resilient to failure, in that they can lose as many n-1 processors with an increase of at most 1 in the distance between any two remaining processors.

III. TYPE II N-CUBE NETWORKS

Since Type II n-cube networks use edges of the n-cube to represent processors, the results of the previous section are not directly applicable. Note that to consider the resistance of Type II n-cube networks to processor failures it is necessary to study the properties of paths from one edge of the n-cube to another, when one or more edges have been removed from the graph. To facilitate this analysis, a concept from graph theory known as an edge graph [3] is useful.

Definition 4: For an undirected graph G, the edge-graph of G is a graph whose vertices correspond to the edges of G, with an edge drawn between two vertices of the edge-graph iff the corresponding two edges in G are incident on a common vertex of G.

Notice that in a Type II n-cube network two processors are distance m apart (i.e. a message from one to the other must traverse m buses) iff in the edge-graph of the n-cube there is a path of length m between the two vertices corresponding to these two processors. Thus the failure resiliency of Type II n-cube networks can be studied by analyzing the connectivity, and r-deleted diameter properties of the edge-graph of the n-cube.

The edge graph of a 3-cube is shown in Figure 1. Since the addresses of the two vertices incident on an edge of the n-cube differ in a single bit, the edge (and corresponding vertex of the edge graph) can be labeled with an address which is equal to the addresses of these vertices with a "-" used in the position in which the addresses differ.

The following theorems give connectivity and r-deleted diameter results for the edge-graph of the n-cube.

Theorem 3: Between any two vertices of the edge graph of the n-cube, whose addresses differ in d positions* ($1 \leq d \leq n-1$), there are 2n-2 vertex disjoint paths, 2d of which are of length d+1, and 2(n-d-1) of which are of length d+3.

Proof: This can be shown constructively. Proof is somewhat lengthy, and is omitted.

Q.E.D.

Theorem 4: The connectivity of the edge-graph of the n-cube is 2n-2 and the r-deleted diameter is n for $0 \leq r < 2n-3$, and n+1 for r = 2n-3.

Proof: Follows directly from Theorem 3.

Q.E.D.

The above result indicates that Type II n-cube networks remain connected even in the presence of as many as 2n-3 (out of $n\,2^{n-1}$) processors, and even in such a situation, interprocessor distances only go up by at most 1.

The results of this section indicate that the edge-graph of the n-cube has some interesting graph-theoretic properties, including very high connectivity with very low deleted diameters. This suggests that for certain applications where high network connectivity is of paramount importance, the edge-graph of the n-cube might provide a good interconnection structure for Type I networks. For example, consider a Type I network of 1024 nodes. If the network were interconnected via a 10-cube structure, it would have a connectivity of 10, an r-deleted diameter of 8 for $0 \leq r < 9$ and a 9-deleted diameter of 11. On the other hand a 1024 node Type I system could employ the edge graph of the 8-cube as an interconnection structure and, by doing so, would achieve a connectivity of 14, an r-deleted diameter of 8 for $0 \leq r < 13$ and a 13-deleted diameter of 9. Of course the price paid for this increase in connectivity and decrease in diameter would be an increase in the number of communication links required, i.e. the latter interconnection structure requires 14 communication links incident on each processor while the former uses only 10.

---

* By definition, two addresses do not differ in any position in which at least one is "-".

IV. ROUTING IN TYPE II N-CUBE NETWORKS

In this section we show that message routing in Type II n-cube networks (and in Type I networks using the edge-graph of the n-cube) is nearly as easy as it is in the n-cube itself, although a slightly more complex routing tag is required. The details of the routing algorithm are as follows:

When a processor $A = (a_1 a_2 \ldots a_n)$ wants to send a message to processor $B = (b_1 b_2 \ldots b_n)$ it tacks onto the message a routing tag of the form $R = (r_1 r_2 \ldots r_n)$ where $r_i \in \{0, 1, '+', 'x'\}$, $1 \leq i \leq n$, according to the following rules:

$$r_i = \begin{cases} '+' & \text{if } b_i = '-' \\ 'x' & \text{if } a_i = b_i \in \{0,1\} \\ 1 & \text{if } a_i \in \{0, '-'\},\ b_i = 1 \\ 0 & \text{if } a_i \in \{1, '-'\},\ b_i = 0 \end{cases}$$

(Note that R will contain a single '+')

Then any time a node $C = (C_1 C_2 \ldots C_{i-1} - C_{i+1} \ldots C_n)$ (including the originator) receives a message for possible forwarding, it executes the following algorithm.

```
If $r_i$ = '+'
Then if all other positions of R are 'x'
    then message is for this node.
    else pass message on to either neighbor whose address has a '-'
        in some position j such that $r_j \in \{0,1\}$.
else begin {$r_i$ must be 0 or 1}
    t = $r_i$;
    set $r_i$ to 'x';
    If all positions of R have value 'x', (except the single
        position which is '+')
    then pass the message on to the neighbor which has position i of
        its address equal to t, and has '-' in the position where
        R has '+'.
    else pass the message on to any neighbor which has position i of
        its address equal to t and has a '-' in any position j such
        that $r_j$ {0,1}.
end
```

This algorithm will always route a message from source to destination via the shortest possible path i.e. a path of length one greater than the number of positions in which the source and destination differ (recall, addresses do not differ in a position in which one or both have value '-').

As an example, Figure 2 shows how the algorithm would route a message between two nodes 0110- and 0011- of the edge graph of the 5-cube.

V. CONCLUSIONS

In this paper we have studied properties of two types of n-cube based networks. Type I networks have been shown to be n-connected, with an r-deleted diameter of n for $0 \leq r < n-1$, and an n-1 deleted diameter of n+1. Type II n-cube networks have been shown to be (2n-2)-connected and to possess an r-deleted diameter of n for $0 \leq r < 2n-3$, and a (2n-3)-deleted diameter of n+1. The results for Type II n-cube networks have led to proposing the use of the edge-graph of the n-cube as a potential interconnection structure for Type I networks. Finally, an algorithm for message routing in Type II n-cube networks (and Type I network based on the edge-graph of the n-cube) has been presented.

REFERENCES

1. H. Sullivan, T. Bashkov and D. Klappholz, "A large scale homogeneous fully distributed parallel machine," in Proc. Fourth Annual Symposium on Computer Architecture, pp. 105-124, March 1977.
2. R.A. Finkel and M.H. Solomon, "Processor interconnection strategies," IEEE Trans. Comput., vol. C-29, pp. 360-371, May 1980.
3. J.A. Bondy and W.S.R. Murty, Graph theory with applications, American Elsevier Publishing Co., 1977.
4. J.G. Kuhl and S.M. Reddy, "Distributed fault-tolerance for large multiprocessor systems," in Proc. of Seventh Annual International Symposium on Computer Architecture, La Baule, France, May 1980.
5. J.G. Kuhl, "fault-diagnosis in computing networks," Technical Report, Dept. of Elec. and Computer Eng., Univ. of Iowa, August 1980.
6. J.R. Armstrong and F.G. Gray, "fault-diagnosis in a Boolean n-cube array of microprocessors", IEEE Trans. Comput. Vol. C-30, pp. 587-590,

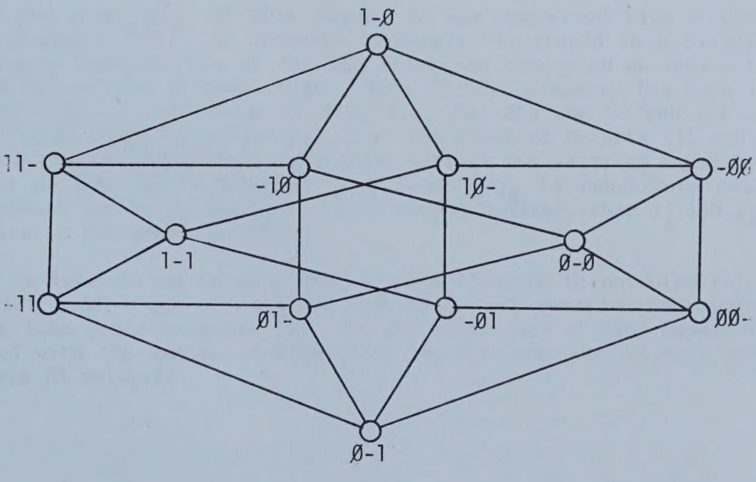


Figure 1. The edge-graph of the binary 3-cube.

address of node receiving message	routing tag as received
0110-	x0x1+
0-100	x0x1+
001-0	xxx1+
0011-	xxxx+

Figure 2. Possible routing of a message in the edge-graph of the 5-cube, as provided by the algorithm of Section IV.

ON GRAPH THEORETIC APPROACH TO n- AND (2n-1) STAGE INTERCONNECTION NETWORKS

DHARMA P. AGRAWAL
Electrical and Computer Engineering, Wayne State University
Detroit, Michigan 48202

ABSTRACT

This paper introduces two graph theoretic models of Multistage Interconnection (MI) networks to show that all nonredundant MI networks with specific buddy property, are topologically equivalent. These models also help in comparing the permutation capabilities of the MI networks with other link-controlled networks like the Augmented Data Manipulator (ADM). These results are used in merging and interconnecting any two strictly buddy-type MI networks or any one MI network and its inverse and thereby making the resultant network topologically equivalent to the Benes network.

I. INTRODUCTION

Nonblocking interprocessor communication in a multiple-processor system, is usually provided by cross-bar switches. A cross-bar switch for m-input m-output case requires m^2 switches and for larger values of m, it is not economical to utilize cross-bar switches. Instead, it is usually divided into several stages of a Multistage Interconnection (MI) network. Though these networks require fewer cross-points and simpler control schemes, these are blocking in nature. The current trend in designing MI networks is to utilize 2-input 2-output switch as a basic building block (Fig. 1). Various such networks have been described in the literature [1-4]. All these network utilize some sort of shuffle and exchange first introduced by Stone [5]. Recently, a comparison of various interconnection networks, has been illustrated by Siegel [6] and Wu and Feng [3] have used the topology describing rules to show the equivalence of various networks. Abidi et al. [7-8] have described single pass conflict free and two single pass conflict-free permutations for various networks.

In the first part of this paper, we are concerned with a class of N-input N-output ($N \equiv 2^n$) MI networks uniformly structured in n stages such that outputs from two SEs of the ith stage are connected as inputs to the same two SEs of the (i+1)th stage. This "Buddy" property has been illustrated in Fig. 2. As shown in Fig. 2.a, two SEs can be buddies in the sense of either inputs (j_1 and j_2) or in the sense of outputs (j_3 and j_4 of Fig. 2.a) and the interstage connection between two pairs of input buddies is shown in Fig. 2.b. Assuming j_1-j_2 and j_3-j_4 to constitute pairs of input buddies and if j_1 and j_3 are also output buddies, then j_2 and j_4 must be buddies in the sense of output.

It is interesting to note that a large class of MI networks fall into the strict buddy type of category and we do not consider non-buddy type networks like Delta networks [9]. In the later part of this paper, we are concerned with the design of Benes-like network from one or more strictly buddy-type MI networks.

II. GRAPH MODELS OF 2x2 SWITCHING ELEMENT (SE)

First Model

A graph model is obtained by representing an SE by a node and the directed edges are drawn starting from the inputs as the source nodes and the outputs as the sink nodes (Fig. 3.a). The control line is totally eliminated from the graph and interchanging the physical positions of x_1 and x_2 and/or y_1 and y_2 does not change the graph model of the SE.

Second Model

An alternative approach is to model the SE based on the connectivity (Fig. 3.b). As the control signal of the SE could be set to either 0 or 1, a node should be approached by only one node and along only one of the two edges. When C=0, the output nodes are approached along the parallel edges and C=1 allows use of cross-connected edges to arrive at the output nodes.

It is worth mentioning that the graph models of Figs. 3.a and 3.b can as well represent the upper (lower) broadcast type of SE introduced by Lawrie for his Omega Network [1].

III. NETWORK MODELING AND THEIR TOPOLOGICAL EQUIVALENCE

The graph model of the SE shown in Fig. 3.a could be easily used to obtain a graph model of a MI network, if the following is assumed:

a. A node is assigned to each SE.
b. Only the primary inputs and the final outputs are assigned nodes.
c. Nodes are not assigned to all intermediate inputs and outputs.
d. Nodes representing SEs of two stages are connected by directed edges.

This procedure leads to a graph model of a network and graph models for 8-inout and 8-output and Baseline [3] network of Fig. 4.a is shown in Fig. 4.b.

The strictly buddy type MI networks described in the literature look different because differing shuffling strategies are utilized for interconnecting the stages of the network. But, mathematical relations have been derived [3] to show that some of the networks are topologically equivalent With our graph model, any MI network can be transformed into directed graphs and then topological equivalence is translated into graphs isomorphism [10].

Theorem 1: All of the 2^n-input 2^n-output non-redundant strictly buddy-type MI networks are topologically equivalent to each other provided they satisfy the following conditions:

a. Each network utilize 2-input 2-output SE as a basic module.
b. Each has n stages.
c. There are 2^{n-1} SEs per stage.
d. For all stages of the network a strict buddy property is observed in both senses of the input and the output.
e. SEs are connected in such a way that any input link can be connected to any one of the output links.

Proof. The theorem is proved using a simplified representation of the graph model as shown in Fig. 5.a. The details inside the rectangular box is not shown to prove the theorem for any general value of n. It is implied that the structure of the graph provides a connecting path from any input node to any one of the output nodes. To prove the theorem, we assume that it is true for Fig. 5.a and we will show its correctness for (2^{n+1}) input network. Such a network can be obtained by having two networks of size 2^n and add one more stage either at input side or the output side.

One such possible configuration with a stage added at the output side is shown in Fig. 5.b. Each pair of nodes satisfying input buddy property, are also marked at nth stage of Fig. 5.b. To provide full connectivity, nodes at nth stage of two 2^n networks, are to be connected togehter at (n+1)th stage. Let us assume that α and γ are connected to a. Then the buddy property requires α and γ to be connected to another same node; say c. Then the output buddy restrictions necessitates that while constructing (n+1) stage, the respective input buddies of α and γ (which are β and δ) ought to be connected to two nodes of (n+1)th stage (say b and d) and there is only one unique way of doing this. This process could be continued for all nodes of nth stage, thus providing only one type of graph. Such buddy type MI network is also referred to as non-redundant as there is a unique path for each input-output pair. Hence, all the strictly buddy type MI networks with full connectivity are topologically equivalent to each other.

Q.E.D.

IV. ALTERNATIVE GRAPH THEORETIC APPROACH AND ITS USEFULNESS

The model of a SE shown in Fig. 3.b could be easily utilized in obtaining an another graph of all MI networks and this alternate model of the Baseline Network is shown in Fig. 6. This model can now be compared with the networks like Augmented Data Manipulator (ADM) [11] and Banyan Networks [12] which do not utilize SEs as basic modules. In ADM, the data output from an individually controlled cell x at stage i becomes the data input to cell x' at state i-1 where x' is defined as:

a. Straight: $x' \equiv x$
b. Upper broadcast: $x' \equiv x+2^i \pmod N$
c. Lower broadcast: $x' \equiv x-2^i \pmod N$

Such an ADM is shown in Fig. 7.a. The second graph model of Fig. 3.b also represents the link connectivity and each cell of ADM could be considered as an input/output node of a SE, thereby providing the graph model of ADM as shown in Fig. 7.b.

Now, the graph models of Figs. 6 and 7.b could be used to compare the permutation capabilities of the Baseline and the ADM. A Baseline network, drawn as a subgraph of the ADM, is shown in Fig. 8 and this shows that the ADM to be topologically equivalent to the Baseline. However, the ADM still contains some unused links as indicated by the broken lines of Fig. 8, which makes the ADM to possess more permutation capabilities as compared to the Baseline or any other MI network as defined by the Theorem 1.

V. BENES TYPE NETWORK IMPLEMENTATION

The Benes Network [13] can pass any input-output bijection in a conflict-free manner. A systematic way of developing such N-input N-output $(N \equiv 2^n)$ network using 2x2 SEs and employing $2\log_2 N-1$ or $(2n-1)$ stages, have been described in [14]. A Benes network for N=8, is shown in Fig. 9.a and its graph using model of Fig. 3.a, is obtained in Fig. 9.b. It might be noted that the strict buddy properties are satisfied by each stage of the Benes network.

A recently described Baseline MI network [3] utilizes a recursive scheme similar to the Benes network and merging the last stage of the Baseline network with the first stage of the Reverse Baseline network gives a Benes network [4]. connectivity property of the graph and the resultant network graph of Fig. 10.b is identical to the Benes network graph of Fig. 9.b.

Theorem 2: A direct combination of n or (n-1) stages of the Baseline and (n-1) or n stages of the Reverse Baseline network (hereafter called Composite Baseline Network) can pass any of the 2^n! possible permutations in a conflict-free manner.

Proof. The graph of Fig. 10.b shows such a combination of the Baseline and the Reverse Baseline networks to exactly identical to the Benes network of Fig. 9.b and the Benes network can provide any one of possible bijections in a conflict-free manner.

Q.E.D.

Corollary 1: The Baseline or the Reverse Baseline networks can be used in either or in both the parts of Fig. 9.

Proof. As shown in [4], the Baseline and the Reverse Baseline networks are not only topologically equivalent but also functionally equivalent to each other. Hence, the two networks can be substituted for each other.

Q.E.D.

Theorem 3: Theorem 2 is valid if the Baseline and the Reverse Baseline networks are replaced by any two MI networks provided:

a. All the conditions of theorem 1 including the strict buddy property are satisfied by the MI networks.
b. Strict buddy property between the two networks is satisfied, i.e. the input buddies of the last stage in the first network are merged with the output buddy pairs of the 1st stage of the second network.

Proof. Just like the Baseline network, two MI networkds can be put together side by side as is shown in Fig. 11.a with the details of the MI networks omitted for the generality of the networks. As per our Theorem 1, the MI networks 1 and 2 are isomorphic and topologically equivalent to the Baseline network. An example is shown in Fig. 12.

Q.E.D.

VI. CONCLUDING REMARKS

The versatility of the proposed technique is shown by a systematic way of proving the topological equivalence of all MI networks with buddy property, by comparing their permutation capabilities with other link-controlled networks and by providing a general design methodology for implementing a Benes-like network from two MI networks. The graph model seems to be very powerful tool and we hope to see its increased use in resolving other aspects of computer networking.

REFERENCES

1. D. Lawrie, "Access and Alignment of data in an array processor", IEEE Transactions on Computers, Vol. C-24, Dec. 1975, pp. 1145-1155.

2. M.C. Pease, "The indirect binary n-cube microprocessor array", IEEE Transactions on Computers, Vol. C-26, May 1977, pp. 458-473.

3. C.L. Wu and T.Y. Feng, "Routing Techniques for a class of multistage interconnection networks", Proceedings of the 1978 International Conference on Parallel Processing, pp. 197-205, also see IEEE Transactions on Computers, Vol. C-29, No. 8, Aug. 1980, pp. 694-702.

4. C.L. Wu and T.Y. Feng, "The Reverse Exchange Interconnection Network", IEEE Transactions on Computers, Vol. C-29, No. 9, Sept. 1980, pp. 801-810.

5. H.S. Stone, "Parallel Processing with the perfect shuffle", IEEE Trans. on Computers, Vol. C-20, No. 2, Feb. 1971, pp. 153-161.

6. H.J. Siegel, "A Model of SIMD Machines and a Comparison of Various Interconnection Networks", IEEE Transactions on Computers, Vol. C-28, December 1979, pp. 907-917.

7. M.A. Abidi and D.P. Agrawal, "On conflict-free permutations in mulitstage interconnection networks", Proceedings of the 1979 International Conference on Parallel Processing, August 1979, also appeared in Journal of Digital Systems, Vol. V, No. 2, Summer 1980, pp. 115-134.

8. M.A. Abidi, D.P. Agrawal and J.J. Metzner, "Two Single Pass Permutations in Multistage Interconnection Networks", presented in the Fourteenth Annual Conference on Information Sciences and Systems, Princeton, March 26-28, 1980, pp. 516-522.

9. D.P. Agrawal and S.C. Kim, "The Theory Underlying the Design of Nonequivalent Multistage Interconnection Networks", presented at 1981 International Conference on Parallel Processing, August 25-28, 1981, pp. 234-237.

10. S. Sahni, "Mathematical Concepts in Science and Engineering", Report, University of Minnesota, Sept. 1980.

11. H.J. Siegel and S.D. Smith, "Study of Multistage SIMD Interconnection Networks", Proc. 5th Annual Symposium on Computer Architecture, April 1978, pp. 223-229.

12. L.R. Goke and G.J. Lipovski, "Banyan networks for partitioning multiprocessor systems," Proceedings First Annual Symposium on Computer Architecture, Dec. 1973, pp. 21-28.

13. V. Benes, Mathematical Theory of Connecting Networks, (Book), New York: Academic, 1965.

14. D. Opferman and N. Tsao-Wu, "On a Class of Rearrangeable Switching Networks", Bell System Technical Journal, Vol. 50, May-June 1971, pp. 1579-1618.

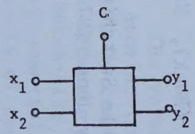
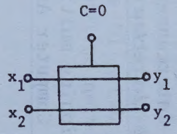
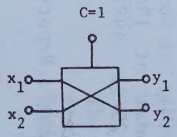
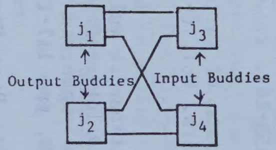
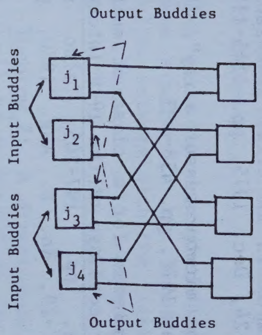


Fig. 1a. Switching Element


C=0
x1
x2
y1
y2
Fig. 1b. Parallel Connection with C=0


C=1
x1
x2
y1
y2
Fig. 1c. Cross-Connection with C=1


j1
j2
j3
j4
Output Buddies
Input Buddies
Fig. 2a. Buddy Property of SEs


Output Buddies
j1
j2
j3
j4
Input Buddies
Input Buddies
Output Buddies
Fig. 2 b. Interstage Buddy Property in a Uniform MI Network

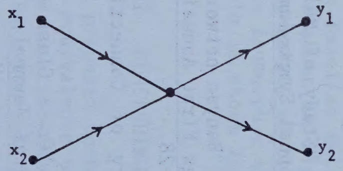


Fig. 3a. Graph Model of Switching Element

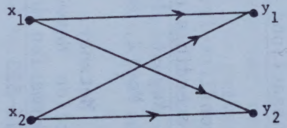
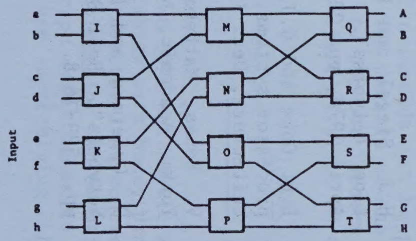
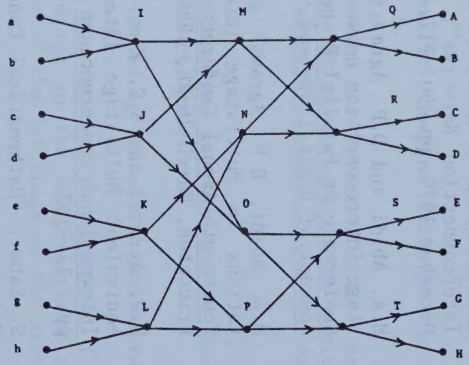
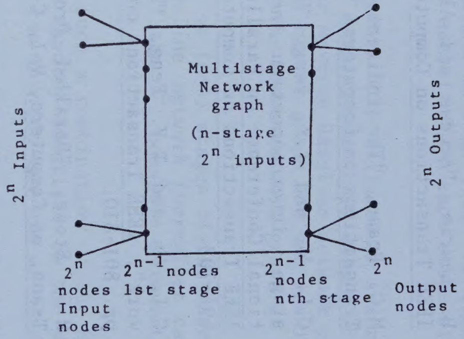


Fig. 3b. Switching Element and its alternative Graph Model


Input
a
b
c
d
e
f
g
h
I
J
K
L
M
N
O
P
Q
R
S
T
A
B
C
D
E
F
G
H
Output
Fig. 4a. Base line network


a
b
c
d
e
f
g
h
I
J
K
L
M
N
O
P
Q
R
S
T
A
B
C
D
E
F
G
H
Fig. 4b. Graph Model of Base-line network


2^n Inputs
Multistage Network graph
(n-stage 2^n inputs)
2^n Outputs
2^n nodes Input nodes
2^{n-1} nodes 1st stage
2^{n-1} nodes nth stage
2^n Output nodes
Fig. 5.a Simplified Representation of n-stage MI network graph model.

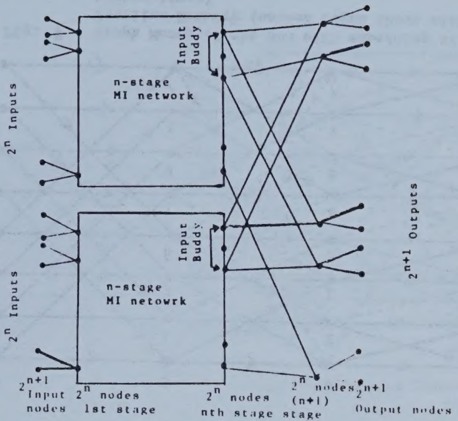

2^n Inputs
n-stage
MI network
Input
Buddy
2^n Inputs
n-stage
MI netowrk
Input
Buddy
2^{n+1} Outputs
2^{n+1}
Input
nodes
2^n nodes
1st stage
2^n nodes
nth stage
2^n nodes
(n+1)
stage
2^{n+1}
Output nodes

Fig. 5.b. Construction of (2^{n+1}) input MI network graph from two networks of size 2^n.

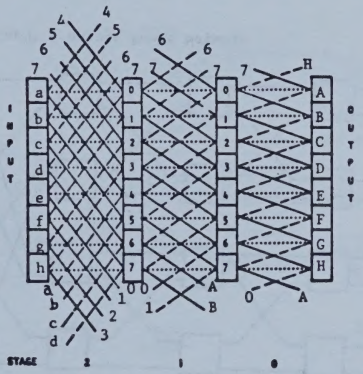

INPUT
OUTPUT
a
b
c
d
e
f
g
h
0
1
2
3
4
5
6
7
A
B
C
D
E
F
G
H
STAGE
2
1
0

Fig. 7a. Augmented Data Manipulator

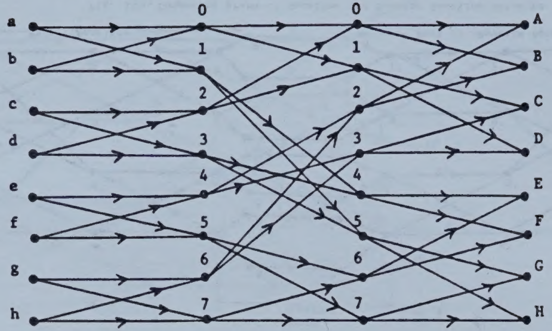

a
b
c
d
e
f
g
h
0
1
2
3
4
5
6
7
A
B
C
D
E
F
G
H

Fig. 6. An Alternative Graph Model of the Baseline Network

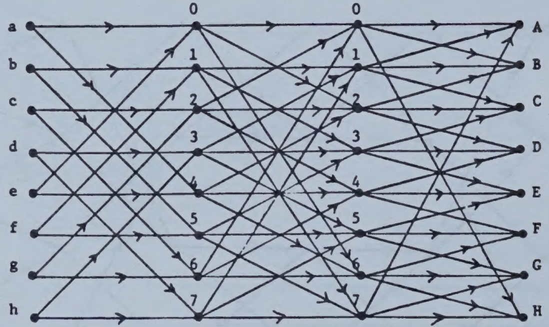

a
b
c
d
e
f
g
h
0
1
2
3
4
5
6
7
A
B
C
D
E
F
G
H

Fig. 7b. Graph Model for the Augmented Data Manipulator (ADM) of Fig.12.a.

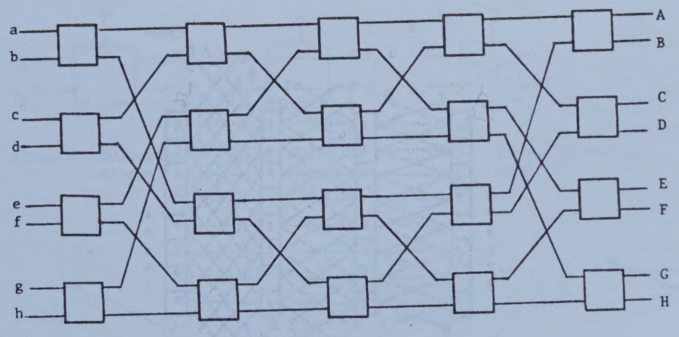

a
b
c
d
e
f
g
h
A
B
C
D
E
F
G
H

Fig. 9a. 8-input 8-output Benes Network

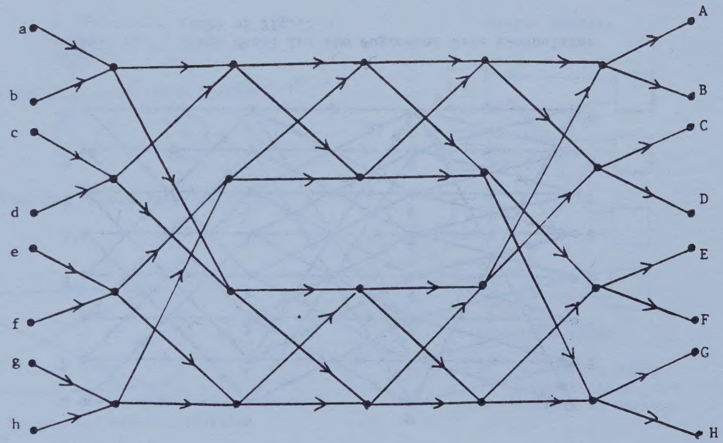

a
b
c
d
e
f
g
h
A
B
C
D
E
F
G
H

Fig. 9b. Graph Model of the Benes Network

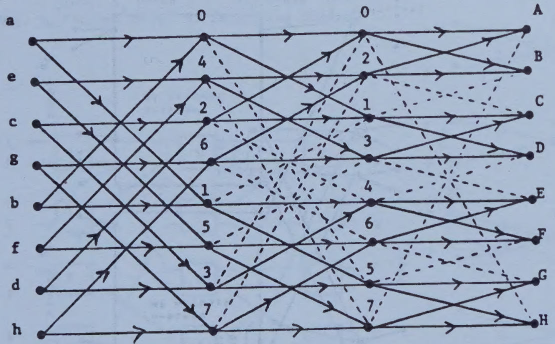

a
e
c
g
b
f
d
h
0
4
2
6
1
5
3
7
0
2
1
3
4
6
5
7
A
B
C
D
E
F
G
H

Fig. 8 . Graph Model of the ADM with embedding of the Baseline Network (unused edges shown with broken lines).

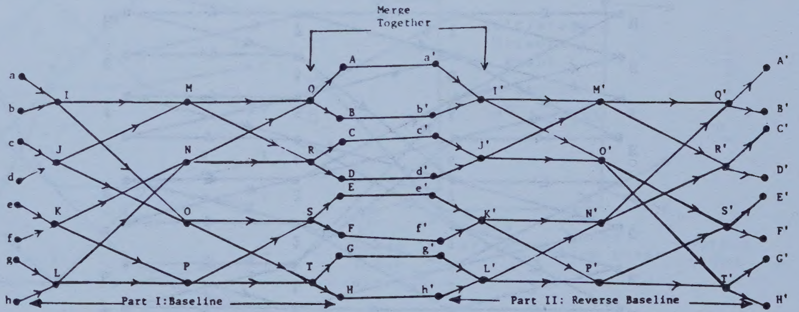

Merge Together
a
b
c
d
e
f
g
h
I
J
K
L
M
N
O
P
Q
R
S
T
A
B
C
D
E
F
G
H
a'
b'
c'
d'
e'
f'
g'
h'
I'
J'
K'
L'
M'
O'
N'
P'
Q'
R'
S'
T'
A'
B'
C'
D'
E'
F'
G'
H'
Part I:Baseline
Part II: Reverse Baseline

Fig. 10a. Composite graph of Baseline and Reverse Baseline Networks

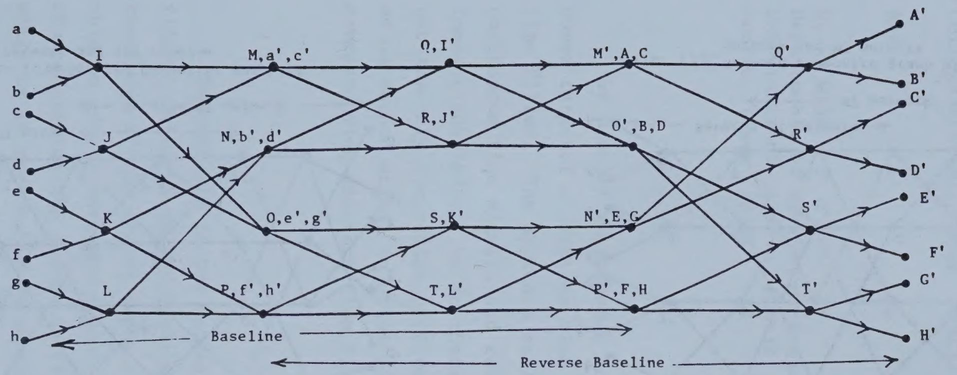

a
b
c
d
e
f
g
h
I
J
K
L
M,a',c'
N,b',d'
O,e',g'
P,f',h'
Q,I'
R,J'
S,K'
T,L'
M',A,C
O',B,D
N',E,G
P',F,H
Q'
R'
S'
T'
A'
B'
C'
D'
E'
F'
G'
H'
Baseline
Reverse Baseline

Fig. 10b. Reduced Composite graph of Baseline and Reverse Baseline Networks

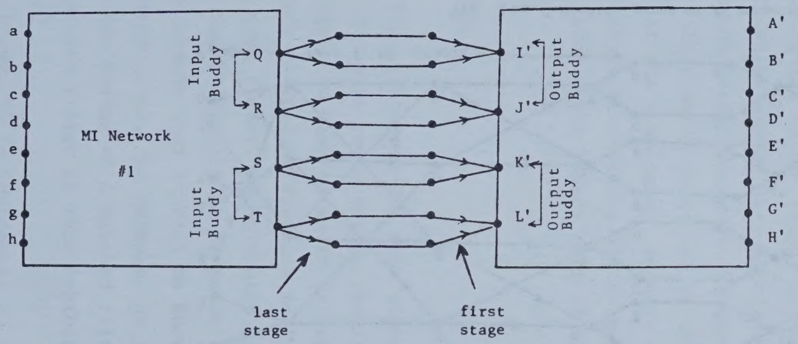

a
b
c
d
e
f
g
h
MI Network
#1
Input Buddy
Q
R
Input Buddy
S
T
I'
J'
Output Buddy
K'
L'
Output Buddy
A'
B'
C'
D'
E'
F'
G'
H'
last stage
first stage

Fig. 11.a Composition of two MI networks

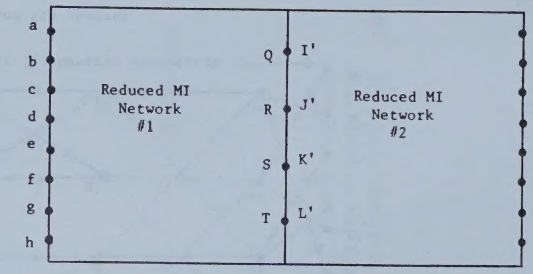

a
b
c
d
e
f
g
h
Reduced MI Network #1
Q
R
S
T
I'
J'
K'
L'
Reduced MI Network #2

Fig. 11.b Reduced Composition of two MI networks

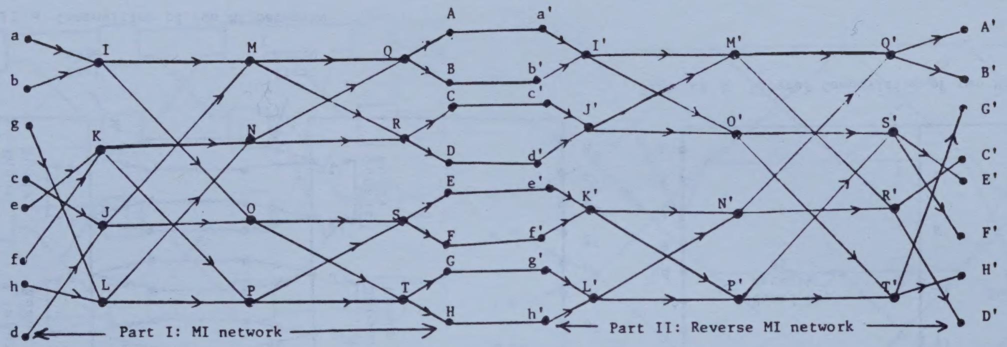

a
b
g
c
e
f
h
d
I
K
J
L
M
N
O
P
Q
R
S
T
A
B
C
D
E
F
G
H
a'
b'
c'
d'
e'
f'
g'
h'
I'
J'
K'
L'
M'
O'
N'
P'
Q'
S'
R'
T'
A'
B'
G'
C'
E'
F'
H'
D'
Part I: MI network
Part II: Reverse MI network

Fig. 12a. Composite graph of MI network and its reverse

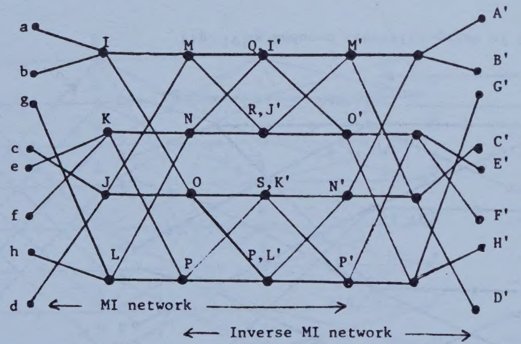

a
b
g
c
e
f
h
d
I
M
Q,I'
M'
K
N
R,J'
O'
J
O
S,K'
N'
L
P
P,L'
P'
A'
B'
G'
C'
E'
F'
H'
D'
MI network
Inverse MI network

Fig. 12b. Reduced Composite graph of MI network and its reverse

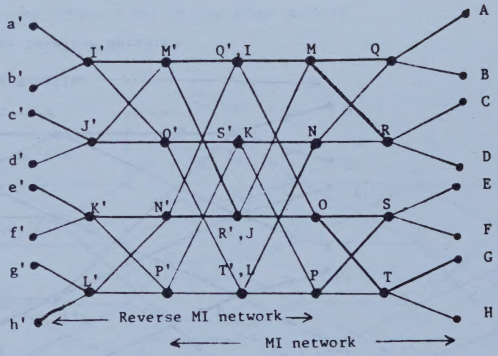

a'
b'
c'
d'
e'
f'
g'
h'
I'
M'
Q',I
M
Q
J'
O'
S',K
N
R
K'
N'
R',J
O
S
L'
P'
T',L
P
T
A
B
C
D
E
F
G
H
Reverse MI network
MI network

Fig. 12c. Reduced Composite graph of Reverse MI network and MI network

AN OBSERVATION CONCERNING SIGNAL TO NOISE RATIO PROPERTIES OF CONTINUOUS AND DISCRETE TIME DETECTORS

D. R. HALVERSON
Department of Electrical Engineering and Applied Research Laboratories
Texas A&M University
College Station, Texas 77843

University of Texas at Austin
Austin, Texas 78712

and

G. L. WISE
Departments of Electrical Engineering and Mathematics
University of Texas at Austin
Austin, Texas 78712

ABSTRACT

We present an approach toward improving the output signal to noise power ratio of a discrete time matched filter by employing a continuous time filter. The approach exploits the bandlimited nature of the unknown continuous time signal, and results in a filter which is robust to this inexact knowledge. We furthermore show that the robustness property does not severely compromise performance; in fact, the resulting filter possesses an output signal to noise power ratio which upper bounds that of the corresponding discrete time filter.

I. INTRODUCTION

The matched filter has been of practical interest for some time. This filter, which maximizes the output signal to noise power ratio, requires knowledge of the signal input for its design. In many cases it is reasonable to expect that the signal will be known at various discrete instants, thus admitting the design of the discrete time filter. Unfortunately, while it is reasonable to expect that the input signal will be known at discrete instants, it is another matter to assume that it will be known exactly as a closed form analytical expression over, for example, an interval of time, which would be necessary for the design of a matched filter in continuous time. Design of the continuous time filter is thus inhibited by such inexact knowledge of the signal input.

While the signal may be incompletely known, it is reasonable to expect that in many cases it could be modeled as bandlimited. If we furthermore assume the signal is known at a fixed number of instants, we might hope that a continuous time filter could be designed which is insensitive to the remaining inexactness in our knowledge of the signal. We also might hope that the resultant filter's performance would be good when compared to that of the discrete time filter.

In this paper we show that an approach realizing these aspirations is possible. We show that the resulting filter is extremely robust to inexact knowledge of the continuous time signal, and simultaneously possesses an output signal to noise power ratio which upper bounds that of the discrete time filter. These results show that in many cases it is desirable to conduct the filtering operation in continuous time, even though the signal may be known imperfectly.

II. DEVELOPMENT

Suppose the continuous time signal s(t) is of finite energy and possesses a Fourier transform which vanishes outside the interval [-W, W], where $W > 0$ is known, and suppose also that $s(t_i) = s_i$ for $i = 1, 2, \ldots, n$. We will employ the filter whose output at time T is $\int y(t)h(T-t)dt$, where y(•) is the received waveform and h(•) is the Green's function of the linear filter (all integrals, unless indicated otherwise, are taken over the entire real line). For the purposes of this discussion we will take T to be fixed and satisfy $T > t_i$ for all $i = 1, 2, \ldots, n$. The output signal to noise power ratio then becomes

$$\mathcal{R}(s, h) = \frac{[\int s(t)h(T-t)dt]^2}{\iint R(t-u)h(T-t)h(T-u)dt\,du},$$

where R(•) is the autocorrelation of the wide sense stationary noise. Note that the finite energy condition on s(•) allows us to place conditions on h(•) independent of s(•) so that $\mathcal{R}(\cdot, \cdot)$ is well defined. We will consider the class $\mathcal{H}$ of all finite energy h(•) such that $0 < \iint R(t-u)h(T-t)h(T-u)\,dt\,du < \infty$. Let $\mathcal{S}$ denote the class of admissable signals, i.e. finite energy signals possessing a Fourier transform which vanishes outside [-W, W] and equaling s_i at t_i, $i = 1, 2, \ldots, n$. We then may interpret the filter design problem as essentially a choice of an element h(•) of $\mathcal{H}$ for a given element s(•) of $\mathcal{S}$.

In practice, of course, we do not know s(•); we only know that it belongs to $\mathcal{S}$. We must therefore accept a certain degree of inexactness of knowledge of s(•) in our decision procedure when choosing h(•). It would obviously be beneficial to make the choice in a way which is insensitive to such inexact knowledge and which leads to good performance. In particular, we might hope that we could choose a filter with performance upper bounding that of the discrete time filter for any $s(\cdot) \in \mathcal{S}$.

The problems imposed by the presence of various types of inexact knowledge have been encountered in many other contexts. Frequently, it has been beneficial to employ approaches which are "robust" to the inexact knowledge. Such methods very often involve saddlepoint criteria, e.g. see [1 - 4]. The general success of this work thus motivates a saddlepoint approach to our somewhat different problem; we therefore seek $s(\cdot)$ and $h(\cdot)$ corresponding to $\inf_{s\in\mathcal{S}} \sup_{h\in\mathcal{H}} \mathcal{R}(s, h)$. Our first result shows how the filter may be designed by this criterion.

Theorem 1: Suppose $\det[(\mathrm{sinc}(W(\cdot-t_i))*R(\cdot))(t_j)]_{i,\,j} \neq 0$, and suppose $R(\cdot)$ has a Fourier transform that is positive on $[-W, W]$. Then we realize $\inf_{s\in\mathcal{S}} \sup_{h\in\mathcal{H}} \mathcal{R}(s, h) = \min_{s\in\mathcal{S}} \max_{h\in\mathcal{H}} \mathcal{R}(s, h)$ via $h(t) = \sum_{i=1}^{n} c_i \, \mathrm{sinc}(W(T-t_i-t))$ and $s(t) = [\sum_{i=1}^{n} c_i \, \mathrm{sinc}(W(\cdot-t_i))*R(\cdot)](t)$, where the constants c_i are chosen so that $s(t_i) = s_i$ for $i = 1, 2, \ldots, n$.

Proof: For any $h\in\mathcal{H}$, let $H(t) \overset{\Delta}{=} h(T-t)$, and denote the Fourier transforms of s, H, R by $\tilde{s}$, $\tilde{H}$, $\tilde{R}$ respectively. Then an argument similar to that of [5] may be used to show that for a fixed s, $\sup_{h\in\mathcal{H}} \mathcal{R}(s, h)$ is realized if and only if $\tilde{s}(\omega) = \tilde{R}(\omega)\,\tilde{H}(\omega)$ for almost all ω. We thus have $\tilde{H}(\omega) = \tilde{s}(\omega)\,\tilde{R}^{-1}(\omega)$ for almost all $\omega \in [-W, W]$. It is routine to verify that $\int s(t)H(t)dt$ is independent of the values $\tilde{H}(\cdot)$ takes on $[-W, W]^c$; moreover it is straightforward to show that $\iint R(t-u)H(t)H(u)dt\,du = \frac{1}{2\pi}\int \tilde{H}(\omega)\tilde{H}(-\omega)\tilde{R}(\omega)d\omega$, which is minimized for $\tilde{H}(\cdot)$ as above by choosing $\tilde{H}(\omega) = 0$ on $[-W, W]^c$. Thus $\sup_{h\in\mathcal{H}} \mathcal{R}(s, h)$ is realized by choosing $\tilde{H}(\omega) = \begin{cases} \tilde{s}(\omega)\;\tilde{R}(\omega)^{-1} & \text{if } \omega \in [-W, W] \\ 0 & \text{otherwise} \end{cases}$. Choosing $h(\cdot)$ corresponding to the above $\tilde{H}(\cdot)$ for a given $s(\cdot)$, we thus must attempt to choose $s(\cdot)$ to realize $\inf_{s\in\mathcal{S}} \sup_{h\in\mathcal{H}} \mathcal{R}(s, h)$. Noting that scaling s allows us to hold the denominator of $\mathcal{R}(s, h)$ constant, we therefore minimize

$$\int s(t)H(t)dt-\lambda\iint H(t)H(u)R(t-u)dt\,du - \sum_{i=1}^{n} \lambda_i \, s(t_i) = \frac{1}{(2\pi)^2}\int\int_{-W}^{W}\int_{-W}^{W} \tilde{s}(\omega)\tilde{\tilde{s}}(\omega)$$

$\tilde{R}^{-1}(\tilde{\omega})e^{j(\omega+\tilde{\omega})t}d\omega\, d\tilde{\omega}\, dt - \frac{\lambda}{(2\pi)^2}\int\int\int_{-W}^{W}\int_{-W}^{W}\tilde{s}(\omega)\tilde{s}(\tilde{\omega})\tilde{R}^{-1}(\omega)\tilde{R}^{-1}(\tilde{\omega})e^{j(\omega t+\tilde{\omega}u)}R(t-u)$ $d\omega\, d\tilde{\omega}\, dt\, du - \frac{1}{2\pi}\sum_{i=1}^{n}\lambda_i\int\tilde{s}(\omega)e^{j\omega t_i}\, d\omega$, where λ and $\{\lambda_i\}_{i=1}^{n}$ are undetermined Lagrange multipliers. Using a variational approach applied to $\tilde{s}(\cdot)$, we minimize $J_{\tilde{s}}(\varepsilon) \overset{\Delta}{=} \frac{1}{(2\pi)^2}\int\int_{-W}^{W}\int_{-W}^{W}(\tilde{s}+\varepsilon\cdot\delta\tilde{s})(\omega)\cdot(\tilde{s}+\varepsilon\cdot\delta\tilde{s})(\tilde{\omega})\tilde{R}^{-1}(\tilde{\omega})e^{j(\omega+\tilde{\omega})t}d\omega\, d\tilde{\omega}\, dt$

$- \frac{\lambda}{(2\pi)^2}\int\int\int_{-W}^{W}\int_{-W}^{W}(\tilde{s}+\varepsilon\cdot\delta\tilde{s})(\omega)\cdot(\tilde{s}+\varepsilon\cdot\delta\tilde{s})(\tilde{\omega})\tilde{R}^{-1}(\omega)\tilde{R}^{-1}(\tilde{\omega})e^{j(\omega t+\tilde{\omega}u)}R(t-u)$

$d\omega\, d\tilde{\omega}\, dt\, du - \frac{1}{2\pi}\sum_{i=1}^{n}\lambda_i\int(\tilde{s}+\varepsilon\cdot\delta\tilde{s})(\omega)e^{j\omega t_i}d\omega$. Letting $J'_{\tilde{s}}(0) = 0$ for all $\delta\tilde{s}$, we obtain $\frac{1}{(2\pi)^2}\int\int_{-W}^{W}\tilde{s}(\tilde{\omega})\tilde{R}^{-1}(\tilde{\omega})e^{j(\omega+\tilde{\omega})t}d\tilde{\omega}\, dt + \frac{1}{(2\pi)^2}\int\int_{-W}^{W}\tilde{s}(\tilde{\omega})\tilde{R}^{-1}(\omega)e^{j(\omega+\tilde{\omega})t}$ $d\tilde{\omega}\, dt - \frac{\lambda}{(2\pi)^2}\int\int\int_{-W}^{W}\tilde{s}(\tilde{\omega})\tilde{R}^{-1}(\tilde{\omega})\tilde{R}^{-1}(\omega)e^{j(\omega t+\tilde{\omega}u)}R(t-u)d\omega\, dt\, du - \frac{\lambda}{(2\pi)^2}\int\int\int_{-W}^{W}\tilde{s}(\tilde{\omega})$ $\cdot\tilde{R}^{-1}(\tilde{\omega})\tilde{R}^{-1}(\omega)e^{j(\tilde{\omega}t+\omega u)}R(t-u)d\tilde{\omega}\, dt\, du - \frac{1}{2\pi}\sum_{i=1}^{n}\lambda_i e^{j\omega t_i} = 0$ for almost all $\omega\in[-W, W]$. This implies via a straightforward calculation $\frac{1}{\pi}\tilde{s}(-\omega)\tilde{R}^{-1}(\omega) - \frac{\lambda}{2\pi}\tilde{s}(-\omega)\tilde{R}^{-1}(\omega) - \frac{1}{2\pi}\sum_{i=1}^{n}\lambda_i\, e^{j\omega t_i} = 0$ for almost all $\omega\in[-W, W]$. Thus $\tilde{s}(\omega) = \sum_{i=1}^{n}\tilde{c}_i\,\tilde{R}(\omega)\, e^{-j\omega t_i}\cdot I_{[-W, W]}(\omega)$ almost everywhere for some constants $\{\tilde{c}_i\}_{i=1}^{n}$, where $I_{[-W, W]}(\cdot)$ is the indicator function for the interval $[-W, W]$. We therefore obtain $s(t) = [\sum_{i=1}^{n} c_i\ \mathrm{sinc}(W(\cdot-t_i))*R(\cdot)](t)$ for some constants $\{c_i\}_{i=1}^{n}$.

In order to establish that this condition is sufficient, we note that a straightforward calculation shows $J_{\tilde{s}}(\varepsilon) = J_{\tilde{s}}(0) + \varepsilon J'_{\tilde{s}}(0) + \frac{\varepsilon^2}{(2\pi)^2}\int\int_{-W}^{W}\int_{-W}^{W}\delta\tilde{s}(\omega)$ $\cdot\delta\tilde{s}(\tilde{\omega})\tilde{R}^{-1}(\tilde{\omega})e^{j(\omega+\tilde{\omega})t}d\omega\, d\tilde{\omega}\, dt - \frac{\varepsilon^2\lambda}{(2\pi)^2}\int\int\int_{-W}^{W}\int_{-W}^{W}\delta\tilde{s}(\omega)\ \delta\tilde{s}(\tilde{\omega})\tilde{R}^{-1}(\omega)\tilde{R}^{-1}(\tilde{\omega})$

$e^{j(\omega t+\tilde{\omega} u)} R(t-u)\, d\omega\, d\tilde{\omega}\, dt\, du \leq J_{\tilde{s}}(0)$ for large positive λ, in view of the hypothesis. Thus the condition $s(t) = [\sum_{i=1}^{n} c_i \operatorname{sinc}(W(\cdot - t_i)) * R(\cdot)](t)$ for some constants $\{c_i\}_{i=1}^{n}$ is both necessary and sufficient. The $\{c_i\}_{i=1}^{n}$ are chosen, of course, to satisfy the constraint $s(t_i) = s_i$ for $i = 1, 2, \ldots, n$, which is possible since $\det[(\operatorname{sinc}(W(\cdot - t_i) * R(\cdot))(t_j)]_{i,j} \neq 0$. Finally we note that $\inf_{s \in \mathcal{S}} \sup_{h \in \mathcal{H}} \mathcal{R}(s, h)$ is realized by choosing h corresponding to the above s, i.e. $\tilde{H}(\omega)$ is almost everywhere given by

$$\tilde{H}(\omega) = \begin{cases} \frac{\pi}{W} \sum_{i=1}^{n} c_i I_{[-W,\, W]}(\omega) e^{-j\omega t_i} & \text{if } \omega \in [-W, W] \\ 0 & \text{otherwise} \end{cases} .$$

Thus $H(t) = \sum_{i=1}^{n} c_i \operatorname{sinc}(W(t-t_i))$, which completes the proof. QED

It might be of interest to determine just how robust the filter of Theorem 1 is. We do know the filter is robust in the minimax sense, but we might in practice desire further knowledge of the sensitivity of the performance of the filter to perturbations in the imperfectly known signal. This is addressed by the following result.

Theorem 2: Let h(•) be the filter of Theorem 1. Then $\mathcal{R}(s, h)$ is independent of $s \in \mathcal{S}$.

Proof: Note that it suffices to show $\int s(t)h(T-t)dt$ is independent of $s \in \mathcal{S}$. Now for each i, $\int s(t) \operatorname{sinc}(W(t-t_i))dt = \frac{1}{4W\pi} \iint_{-W}^{W} \tilde{s}(\omega) e^{j\omega t} d\omega \int_{-W}^{W} e^{(-t_i\tilde{\omega}+\tilde{\omega}t)j} d\tilde{\omega}\, dt = \frac{1}{2W} \int_{-W}^{W} \tilde{s}(\omega)\, e^{jt_i\omega}\, d\omega = \frac{\pi}{W} s(t_i) = \frac{\pi}{W} s_i$. Thus, $\int s(t)h(T-t)dt = \frac{\pi}{W} \sum_{i=1}^{n} c_i s_i$, which establishes the desired result. QED

We therefore can see that in view of Theorem 2 the filter of Theorem 1 is indeed very robust to inexact knowledge of the signal s(•), as long as $s(\cdot) \in \mathcal{S}$. This filter therefore accomplishes our goal of desensitizing the performance of the filter to inexact knowledge of the signal. The other goal was to accomplish the desensitization in a manner which resulted in good performance; in particular, we would hope that the performance would be at least as good (and often better) than that of the discrete time filter. This is illustrated by the following result:

Theorem 3: Suppose $R(\cdot)$ is differentiable in a neighborhood of t_i for each i. Suppose the output signal to noise power ratio of the discrete time matched filter exists and is denoted by $\mathcal{R}_0$, and let $h(\cdot)$ denote the (continuous time) filter of Theorem 1. Then $\mathcal{R}(s, h) \geq \mathcal{R}_0$ for all $s \in \mathcal{S}$.

Proof: For each positive real number Δt, let $P_{i,\Delta t}(t) = \frac{1}{\Delta t} \cdot I_{[t_i, t_i+\Delta t]}(t)$. Throughout the remainder of this proof assume Δt is small enough so that for each i, $[t_i, t_i+\Delta t]$ is contained in the neighborhood of t_i for which $R(\cdot)$ is differentiable. Note that in view of Theorem 2, it suffices to show $\mathcal{R}(s, h) \geq \mathcal{R}_0$ for a particular $s \in \mathcal{S}$. Moreover, since $h(\cdot)$ maximizes performance over $\mathcal{H}$ for a particular $s \in \mathcal{S}$, it suffices to show that there exists a sequence of filters $h_K \in \mathcal{H}$ such that $\mathcal{R}(s, h_K) \to \mathcal{R}_0$, where henceforth $s(\cdot)$ corresponds to $h(\cdot)$ according to Theorem 1. Let $h_{\Delta t}(t) = \sum_{i=1}^{n} s_i P_{i,\Delta t}(T-t)$.

Since $h_{\Delta t}(\cdot) \in \mathcal{H}$ for all Δt sufficiently small, it therefore suffices to show $\lim_{\Delta t \to 0} \mathcal{R}(s, h_{\Delta t}) = \mathcal{R}_0$. Now $\int s(t) h_{\Delta t}(T-t)dt = \sum_{i=1}^{n} s_i \int_{t_i}^{t_i+\Delta t} s(t) \cdot \frac{1}{\Delta t}\, dt$,

and it thus follows that $\lim_{\Delta t \to 0} \int s(t) h_{\Delta t}(T-t)dt = \sum_{i=1}^{n} s_i^2$. Moreover, $\iint R(t-u) h_{\Delta t}(T-t) h_{\Delta t}(T-u)dt\, du = \sum_{i=1}^{n}\sum_{j=1}^{n} s_i s_j \int_{t_i}^{t_i+\Delta t}\int_{t_j}^{t_j+\Delta t} R(t-u) \cdot \frac{1}{(\Delta t)^2}\, dt\, du$, and the conditions of the hypothesis allow the use of Leibnitz's rule, which implies

$\lim_{\Delta t \to 0} \iint R(t-u) h_{\Delta t}(T-t) h_{\Delta t}(T-u)dt\, du = \sum_{i=1}^{n}\sum_{j=1}^{n} s_i s_j R(t_i - t_j)$. We therefore conclude $\lim_{\Delta t \to 0} \mathcal{R}(s, h_{\Delta t}) \to \mathcal{R}_0$, which establishes the desired result.

QED

Note that the additional condition for Theorem 3 of $R(\cdot)$ differentiable in a neighborhood of t_i for all i is quite mild. The continuous time filter in fact often outperforms the discrete time filter. For example, if $R(t) = \text{sinc}(\tilde{W}t)$, where $\tilde{W} > W$, n=2, and $t_2 - t_1 = \pi m/\tilde{W}$ for some positive integer m, then it is straightforward to show that the continuous time filter outperforms the discrete time filter for all $(s_1, s_2) \neq (0, 0)$.

III. CONCLUSION

We have introduced a method to improve the output signal to noise power ratio of a discrete time matched filter by employing a continuous time filter. This method exploits the bandlimited nature of the unknown continuous time signal and is robust to this inexact knowledge. We have furthermore showed that the method results in an output signal to noise power ratio which upper bounds that of the discrete time filter.

ACKNOWLEDGEMENT

This research was supported by the Office of Naval Research under Contract N00014-81-K-0145 and by the Air Force Office of Scientific Research under Grant AFOSR-81-0047.

REFERENCES

1. A. H. El-Sawy and V. D. Vandelinde, "Robust detection of known signals," IEEE Trans. Inform. Theory, vol. IT-23, pp. 722-727, November 1977.

2. S. A. Kassam and J. B. Thomas, "Asymptotically robust detection of a known signal in contaminated non-Gaussian noise," IEEE Trans. Inform. Theory, vol. IT-22, pp. 22-26, January 1976.

3. J. M. Morris, "On single-sample robust detection of known signals with additive unknown-mean amplitude-bounded random interference," IEEE Trans. Inform. Theory, vol. IT-26, pp. 199-209, March 1980.

4. H. V. Poor, "Robust decision design using a distance criterion," IEEE Trans. Inform. Theory, vol. IT-26, pp. 575-587, September 1980.

5. J. B. Thomas, An Introduction to Statistical Communication Theory, chapter 5. New York: Wiley, 1969.

FINITE LENGTH DISCRETE MATCHED FILTERS

Andrew B. Martinez and John B. Thomas
Dept. of Electrical Engineering and Computer Science
Princeton University
Princeton, New Jersey 08544

Abstract

The Matched Filter (MF) is well known to be the linear detector that has the maximum output Signal-to-Noise Ratio (*SNR*). The problem of finding the minimum filter length in discrete time to achieve a certain level of performance is considered when there is some freedom in choosing a signal shape. Upper and lower bounds on the *SNR* are given in terms of the eigenvalues of the noise covariance matrix. Since these bounds are rather difficult to compute, looser, but easier to compute bounds are given. Several examples are presented which illustrate the exact and approximate bounds.

I. Introduction

The design and implementation of the Matched Filter (MF) has received considerable attention [4-8]. As a detector it has the advantage of linearity, and since it is based only on easily estimated 2nd order noise statistics, the MF is simple to optimize. The performance criterion, the Signal-to-Noise Ratio (*SNR*) is tractable, and intuitively appealing.

For a fixed signal in discrete time, Levinson [8] has presented a simple and efficient algorithm to solve the MF equation. Since the MF impulse response and the *SNR* are computed iteratively, the algorithm can be terminated when a filter with desired performance is found. Unfortunately, when there is some freedom in choosing a signal, the choice of signal plays an important part in optimizing the detector. Because the optimal signal of length M is a truncated version of the optimal length $M+1$ signal under only very special conditions, the Levinson algorithm must be repeated N times, and thus loses its computational advantage. In this paper, easily computed bounds on the performance of the MF as a function of length are found. Then, before any attempt is made to solve the MF equation, an estimate of the filter length can be found from these bounds.

II. Detection Problem

The detection problem considered in this paper is one of finding a linear detector that discriminates between an hypothesis H_0 and an alternative H_1. The decision is based on a discrete observation vector $\mathbf{x}$ composed under H_0 of noise $\mathbf{n}$ with density f and under H_1 of a known signal $\mathbf{s}$ in noise:

$$H_0: \mathbf{x} = \mathbf{n}$$

$$H_1: \mathbf{x} = \mathbf{n} + \mathbf{s}$$

The detector consists of a real scalar test statistic $T(\mathbf{x})$, a function on $\mathbf{x}$, which is compared to a threshold to decide for H_0 or H_1.

The criterion of optimality used in this paper is a *SNR* measure often called the deflection:

$$SNR = \frac{[E_1(T) - E_0(T)]^2}{Var_0(T)}$$

It is well known that the log likelihood ratio detector for Gaussian noise is linear, the matched filter. Since the detector power is a monotone increasing function of the *SNR* of *T*, the *SNR* is frequently used as a measure of detector performance. Justification of the *SNR*, outside of its intuitive appeal is often made by making a Gaussian assumption about $\mathbf{n}$ or applying the central limit theorem to *T*.

Using the MF as a detector for non-Gaussian noise is more difficult to justify. In general, the likelihood ratio detector maximizes the *SNR* [1], and by a simple calculus of variations argument, maximizing the *SNR* (as defined above) with no restriction on the linearity of the detector can be shown to yield a linear function of the likelihood ratio. The MF is the linear filter which maximizes output *SNR*, but the likelihood ratio is generally nonlinear. Therefore in making a restriction of linearity on *T*, it is tacitly assumed that the noise is Gaussian or nearly Gaussian in the sense that the MF performs reasonably well and that any loss of optimality is compensated for by the simplicity and linearity of the MF.

Under the assumption of linearity, the test statistic $T(\mathbf{x})$ is equal to the output at time N of a linear filter with impulse response $\mathbf{h}$. As a convenience, the pseudo-signal is defined to be a length N vector with elements $u_i = h_{N-i}$. The output *SNR* of the linear detector is given by

$$SNR_o = (\mathbf{u}^T\mathbf{s})^2/\mathbf{u}^T\mathbf{R}\mathbf{u} \tag{1}$$

where $\mathbf{R}$ is the noise covariance matrix. The SNR has a maximum value of λ for the optimal pseudo-signal, thus

$$SNR_o(\mathbf{u}) = (\mathbf{u}^T\mathbf{s})^2/\mathbf{u}^T\mathbf{R}\mathbf{u} \leqslant \lambda$$

Cross-multiplication yields the equation:

$$L(\mathbf{u}) = (\mathbf{u}^T\mathbf{s})^2 - \lambda\ \mathbf{u}^T\mathbf{R}\mathbf{u} \leqslant 0$$

This can be maximized in the usual way by setting its gradient equal to zero:

$$\nabla L(\mathbf{u}) = 2(\mathbf{u}^T\mathbf{s})\mathbf{s} - 2\lambda\mathbf{u}^T\mathbf{R}\mathbf{u} = 0$$

Rearranging and noting that $\lambda/\mathbf{u}^T\mathbf{s}$ is a constant and can be set equal to one with no loss of generality, the result is the well known MF equation:

$$\mathbf{s} = (\lambda/\mathbf{u}^T\mathbf{s})\mathbf{R}\mathbf{u} = \mathbf{R}\mathbf{u} \tag{2}$$

The solution of Eq. (2) is the pseudo signal of the MF:

$$\mathbf{u} = \mathbf{R}^{-1}\mathbf{s}$$

with output *SNR* given by

$$SNR_o = \mathbf{u}^T\mathbf{s} = \mathbf{s}^T\mathbf{R}^{-1}\mathbf{s}$$

In discrete time and with a fixed signal, the MF matrix equation can be solved quite efficiently using the Levinson algorithm. In continuous time, the classical method of solution is to use spectral factorization to solve the equation on an infinite interval; this (possibly) non-causal solution is then projected onto a causal space [3]. In discrete time, there is a parallel spectral approach using the eigenvectors and values of $\mathbf{R}$. This method is discussed in the next section.

III. Signal Selection and Bounds on the *SNR*

It is well known that the MF is the linear filter with the maximum SNR_o for a given signal in noise. In addition for non-white noise, the SNR_o of the MF can be maximized by proper choice of signal shape. Because of this, for signals of constant energy, the SNR_o of the MF has a range of possible values.

Since the $N \times N$ covariance matrix $\mathbf{R}$ is positive definite and Hermitian, it has positive, real eigenvalues:

$$0 < \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_N$$

and a corresponding set of orthonormal eigenvectors:

$$\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N$$

The matrix $\mathbf{R}$ can be diagonalized:

$$\mathbf{R} = \mathbf{E} \Lambda \mathbf{E}^{-1}$$

where $\mathbf{E}$ is the eigenvector matrix:

$$\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N]$$

and Λ is a matrix of eigenvalues:

$$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix}$$

Thus the MF equation has the solution:

$$\mathbf{u} = \mathbf{E} \Lambda^{-1} \mathbf{E}^{-1} \mathbf{s} \tag{3}$$

where Λ^{-1} has as diagonal elements the eigenvalue of $\mathbf{R}^{-1}$

$$\frac{1}{\lambda_1} \geqslant \frac{1}{\lambda_2} \geqslant \cdots \geqslant \frac{1}{\lambda_N}$$

The signal $\mathbf{s}$ can be expanded in terms of the eigenvectors:

$$\mathbf{s} = \mathbf{E}\mathbf{c} \tag{4}$$

where $\mathbf{c}$ is the coordinate of $\mathbf{s}$ under the basis formed by the orthonormal eigenvectors of $\mathbf{R}$. From Eqs. (3) and (4) the pseudo-signal is

$$\mathbf{u} = \mathbf{E} \Lambda^{-1} \mathbf{c} \tag{5}$$

The SNR_o of the MF becomes

$$SNR_o = \mathbf{c}^T \Lambda^{-1} \mathbf{c}$$

If the signal is chosen to be in the eigenspace of the ith eigenvector ($\mathbf{s} = k\mathbf{e}_i$), then

$$SNR_o = k^2 / \lambda_i$$

The Rayleigh quotient theorem [2] states that

$$\frac{1}{\lambda_N} \leqslant \frac{\mathbf{s}^T \mathbf{R}^{-1} \mathbf{s}}{\mathbf{s}^T \mathbf{s}} \leqslant \frac{1}{\lambda_1}$$

where the upper and lower bounds are achieved for signals in the eigenspace of $\mathbf{e}_1$ and $\mathbf{e}_N$ respectively. Thus the MF SNR_o is bounded:

$$\frac{\mathbf{s}^T\mathbf{s}}{\lambda_N} \leqslant SNR_o \leqslant \frac{\mathbf{s}^T\mathbf{s}}{\lambda_1} \tag{6}$$

The best choice of signal is $k\mathbf{e}_1$, the eigenvector of $\mathbf{R}$ with the smallest eigenvalue. This is equivalent to putting the signal in that part of the spectrum of $\mathbf{R}$ where the noise has the smallest magnitude.

A minimax strategy is used by Turin [5] to find the worst-case noise, and the corresponding best signals in continuous time. He shows that the best signal spectrum should consist of the noise spectral components with the smallest magnitude. As a consequence, the worst spectra and the best signal both have flat spectra.

Grettenberg [7] has taken the logical step of using M eigenvectors as an M character alphabet of signals. By choosing the eigenvectors of $\mathbf{R}$ corresponding to the smallest M eigenvalues, not only is the set orthogonal, but it achieves the greatest minimum SNR_o of any such M character set. This also has an advantage of simplicity, since, when the signal is chosen to be an eigenvector of $\mathbf{R}$, from Eq. (9) the pseudo-signal equals the signal, and the MF reduces to a simple correlator.

For the Levinson algorithm to produce the s-optimal MF on each iteration, the length N optimal eigenvector has to be a truncated version of the length $N+1$ eigenvector. Then

$$\mathbf{R}\mathbf{e}_1^{(N+1)} = \lambda_1^{(N+1)}\,\mathbf{e}_1^{(N+1)} = \begin{bmatrix} \lambda_1^{(N)}\mathbf{e}_1^{(N)} \\ \tilde{\mathbf{r}}^T\mathbf{e}_1^{(N+1)} \end{bmatrix} + (\mathbf{e}_1^{(N+1)})_{N+1}\,\tilde{\mathbf{r}}$$

where

$$\tilde{\mathbf{r}} = [r_N \; r_{N-1} \; \cdots \; 0.5r_0]^T$$

and $(\mathbf{e}_1^{(N+1)})_{N+1}$ is the $N+1$st element of $\mathbf{e}_1^{(N+1)}$. For $\mathbf{e}_1^{(N)}$ to be a truncated version of $\mathbf{e}_1^{(N+1)}$, then

$$(\mathbf{e}_1^{(N)})_i = \frac{(\mathbf{e}_1^{(N+1)})_{N+1}}{(\lambda_1^{(N+1)} - \lambda_1^{(N)})}\, r_{N+1-i}$$

Solving iteratively, either $r_i = 0$ or $r_i = (sgn\,(r_1))^{i+1}$ for $i > 0$.

IV. Approximate Bounds on *SNR*

It is impractical to find a suitable filter length N from the bounds in Eq. (6) since they require knowledge of each $M \times M$ minor of $\mathbf{R}$. Looser but easier to compute bounds can be found.

The equivalent rectangular time duration ΔT of the noise autocovariance is introduced as a rough measure of correlation [3]:

$$\Delta T = \sum_{i=-\infty}^{\infty} |r_i|/\sigma^2$$

The largest eigenvalue of $\mathbf{R}$, λ_N is well known to be the smallest norm of $\mathbf{R}$, thus using another norm:

$$\lambda_N \leqslant \|\mathbf{R}\|_\infty = \max_i \sum_j r_{ij} \leqslant \sigma^2 \Delta T$$

This yields the looser bound:

$$\frac{\mathbf{s}^T\mathbf{s}}{\sigma^2 \Delta T} \leqslant \frac{\mathbf{s}^T\mathbf{s}}{\max\limits_i \sum\limits_j r_{ij}} \leqslant \frac{\mathbf{s}^T\mathbf{s}}{\lambda_N} \leqslant SNR_o$$

An upper bound can be found. The condition number K of a matrix is defined as:

$$K = \lambda_{\max}/\lambda_{\min} = \|\mathbf{R}^{-1}\| \, \|\mathbf{R}\|$$

then from Eq. (6)

$$SNR \leqslant \mathbf{s}^T\mathbf{s}\; K/\lambda_{\max}$$

The trace of $\mathbf{R}$ equals the sum of its eigenvalues:

$$tr(\mathbf{R}) = N\sigma^2 = \sum_{i=1}^{N} \lambda_i$$

therefore

$$\sigma^2 \leqslant \lambda_{\max} \leqslant N\sigma^2$$

and so

$$\frac{\mathbf{s}^T\mathbf{s}}{\sigma^2 \Delta T} \leqslant SNR_o \leqslant \mathbf{s}^T\mathbf{s}\; K/\sigma^2 \tag{7}$$

Since the input SNR is given by

$$SNR_i = \mathbf{s}^T\mathbf{s}/N\sigma^2$$

The improvement in SNR of the MF is given by

$$\frac{N}{\Delta T} \leqslant SNR_o/SNR_i = SNR_{MF} \leqslant NK \tag{8}$$

The Time-Bandwidth product (TW) has frequently been used as a measure of both detector performance and filter complexity [4-6]. The input to the detector is assumed to be bandlimited with single-sided equivalent rectangular bandwidth W. Thus the input noise variance is $2W\sigma^2$ and the signal has power

$$P_s = \frac{1}{T}\int s^2(t)\, dt$$

The input SNR is

$$SNR_i = P_s/2TW\sigma^2$$

The MF is designed for non-bandlimited white noise with covariance $\mathbf{R} = \sigma^2\mathbf{I}$. Thus the output SNR is

$$SNR_o = P_s/\sigma^2$$

and the improvement SNR_{MF} of the MF is

$$SNR_{MF} = SNR_o/SNR_i = 2TW$$

Turin [5] has shown that maximizing the TW is equivalent to maximizing the SNR_o. Although for true white noise the SNR_o depends only on the energy in the signal, for power limited interference (jamming), large TW signals are still best.

V. Examples

As a first example, consider white noise with an $N \times N$ covariance matrix $\mathbf{R} = \sigma^2 \mathbf{I}$. The covariance matrix has an Nth order eigenvalue σ^2 making the upper and lower bounds in Eq. (6) equal, and the choice of signal arbitrary. Other considerations, such as a ceiling on transmitted signal strength, may still make the spreading of signal energy in time desirable.

For $N = 3$, the autocorrelation sequence is given by

$$\mathbf{r} = [1 \; r_1 \; r_2]^T$$

For $\mathbf{R}$ to be positive definite, the values that r_1 and r_2 can take are restricted:

$$|r_1| \leqslant 1 \qquad \text{and} \qquad r_2 \leqslant \sqrt{(r_2+1)/2}$$

This region of the $r_1 \times r_2$ plane is shown in Figs. 1 and 2.

The difference in dB between the upper and lower *SNR* bounds in Eq. (6) is plotted as contours in Fig. 1. Even for a filter this short, the signal selection is shown to be quite important.

In Fig. 2, the *SNR* of the MF for an alternating signal ($s_i = (-1)^i$) is shown in dB over the lower bound. The alternating signal was chosen as a suboptimal approximation to the optimal signal because of its simplicity, and similarity in shape to the optimal signal for $r_1 \geqslant 0$. It is readily seen to be nearly optimal in this case. Because of the symmetry of this problem, a constant signal ($s_i = 1$), chosen as a suboptimal signal for $r_i < 0$ has performance contours which are the mirror image of those in Fig. 2.

Four noise autocorrelation functions were chosen as representative -- the exponential:

$$r_i = \exp(-0.2\,|i|)$$

the triangular:

$$r_i = \begin{cases} 1 - |i|/10 & |i| \leqslant 10 \\ 0 & |i| > 10 \end{cases}$$

the Gaussian:

$$r_i = \exp(-\pi\,(i/10)^2)$$

and the hyperbolic secant:

$$r_i = \mathit{sech}\,(\pi\, i/10)$$

The exponential is the simplest member of the Markov class; the triangular function has finite support; the Gaussian correlation function has infinite support, yet has tails which fall off faster than the exponential, and the hyperbolic secant has a nearly Gaussian shape at the origin, but exponential tails.

The upper and lower bounds on the SNR_{MF} from Eq. (6) are plotted in dB versus filter length N in Figs. 3-6. Here signal selection is extremely important for all $N > 2$ and increasingly so for increasing N. Even for the length 5 filter, the difference between the best and worse-case SNR_{MF} is at least 15dB for all four cases. At $N = 20$, the difference is at least 19dB.

The parameters of these four correlation functions were chosen so that each has an equivalent rectangular time duration of $\Delta T \approx 10$. Thus the approximate lower bound (Eq. 8) for each function is

$$N/10 \leqslant SNR_{MF}$$

The approximate and exact lower bounds for each correlation function is shown is Figs. 7-10.

V. Conclusion

The bounds on SNR_{MF} in Eq. (6) show the selection of the signal to be important for optimal performance of the MF. The selection of a suboptimal signal, if made intelligently, can produce nearly optimal results, and certainly the importance of signal shape should not be overlooked.

The approximate lower bound of Eq. (8) gives a simple although conservative estimate of worse case MF performance. An estimate of filter length can be made with only knowledge of the equivalent rectangular time duration ΔT. The approximate upper bound in Eq. (8) requires knowledge of the condition number of the matrix. While difficult to compute, it has some theoretical interest. The condition number, as the ratio of the greatest to smallest spectral component magnitudes, gives an indication of the non-whiteness of the noise spectrum. It is speculated that the NK product may have other interesting properties, and warrants further investigation.

Acknowledgement

This research is supported by the Office of Naval Research under Contract N00014-81-K-0146 and by the National Science Foundation under Grant ECS-79-18915.

References

[1]. C.W. Helstrom, *Statistical Theory of Signal Detection.* Oxford:Pergamon Press, 1968.

[2]. H.R. Schwartz, H. Rutishauser, and E. Stiefel, *Numerical Analysis of Symmetric Matrices.* New York:Prentice Hall, 1973.

[3]. J.B. Thomas, *Introduction to Statistical Communication Theory.* New York:Wiley, 1969.

[4]. G.L. Turin, "An Introduction to Matched Filters," *IRE Trans. on Information Theory,* Vol. IT-6, June 1960, pp. 311-329.

[5]. G.L. Turin, "Minimax Strategies for Matched Filter Detection," *IEEE Trans. on Communications,* Vol. COM-23, No. 11, Nov. 1975, pp. 1370-1371.

[6]. G.L. Turin, "An Introduction to Digital Matched Filters," *Proc. of IEEE,* Vol. 64, No. 7, July 1976, pp. 1092-1112.

[7]. T.L. Grettenberg, "Signal Selection in Communication and Radar Systems" *IEEE Trans. on Information Theory,* Vol. IT-9, No. 4, Oct. 1963, pp. 265-275.

[8]. N. Levinson, "The Wiener RMS (Root Mean Square) Error Criterion in Filter Design and Prediction," *Journal of Mathematics and Physics,* Vol. XXV, No. 4, Jan. 1947, pp. 261-278.

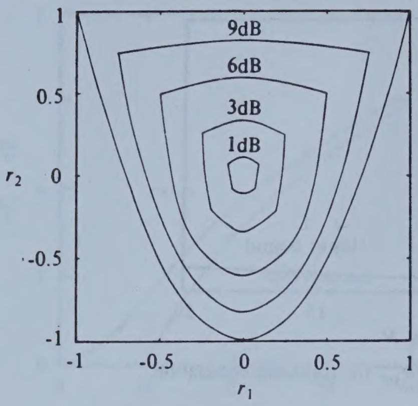


Fig. 1. Contours of SNR_{MF} upper bound for $N=3$ in $r_1 \times r_2$ plane.

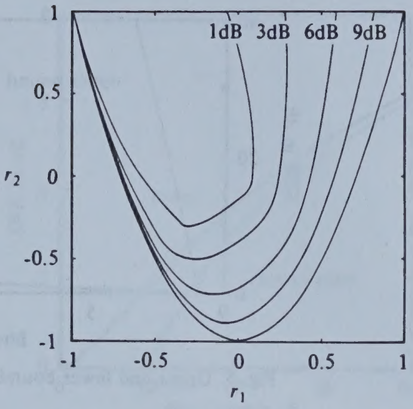


Fig. 2. Contours of SNR_{MF} for alternating signal in $r_1 \times r_2$ plane.

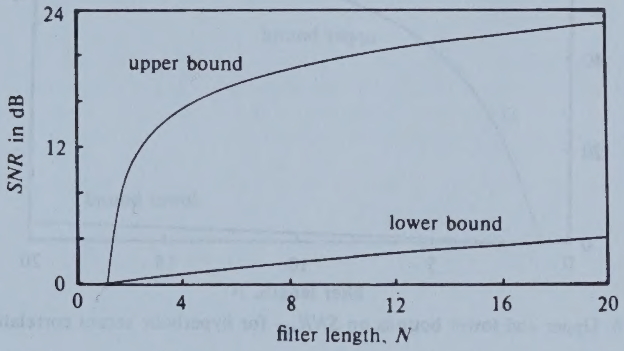


Fig. 3. Upper and lower bounds on SNR_{MF} for exponential correlation.

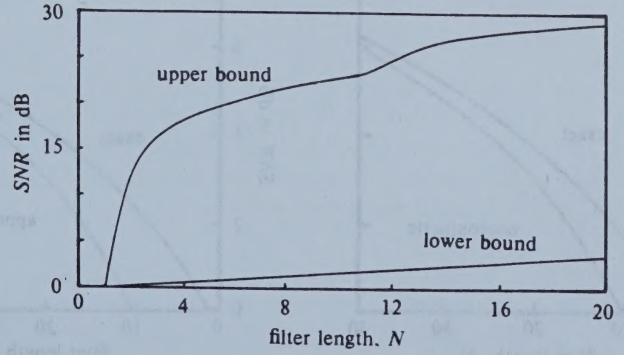


Fig. 4. Upper and lower bounds on SNR_{MF} for triangular correlation.

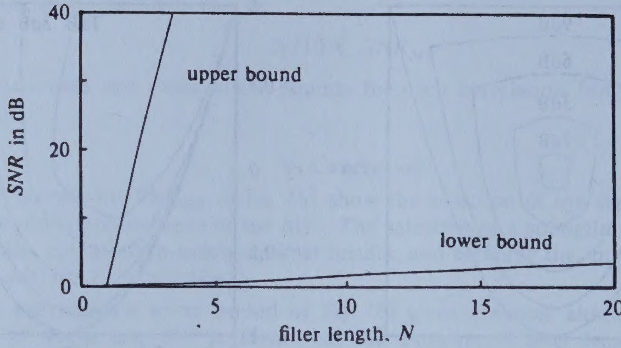


Fig. 5. Upper and lower bounds on SNR_{MF} for Gaussian correlation.

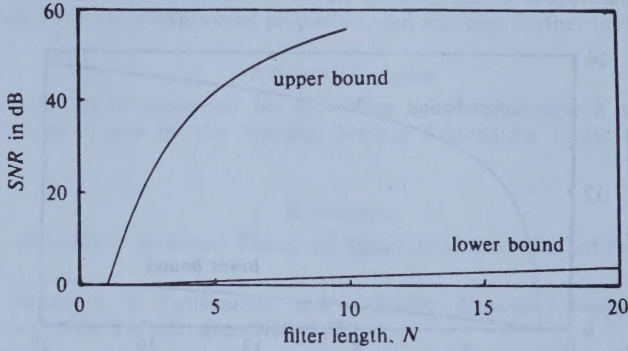


Fig. 6. Upper and lower bounds on SNR_{MF} for hyperbolic secant correlation.

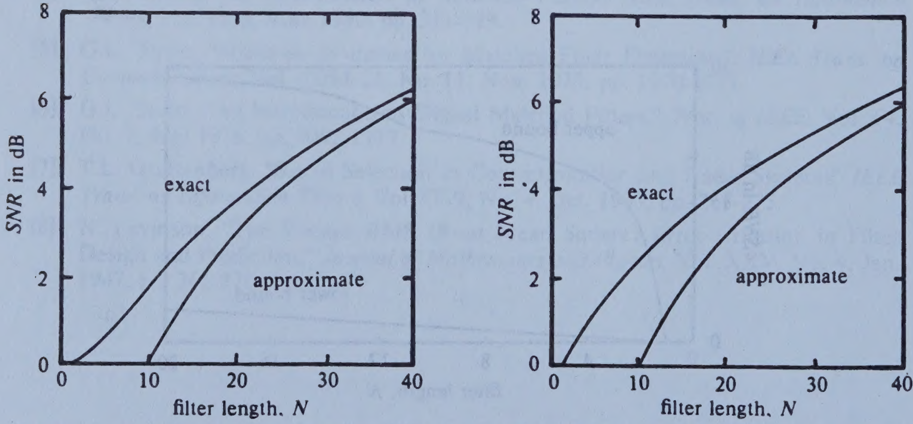


Fig. 7. Approximate and exact lower bound for exponential correlation.

Fig. 8. Approximate and exact lower bound for triangular correlation.

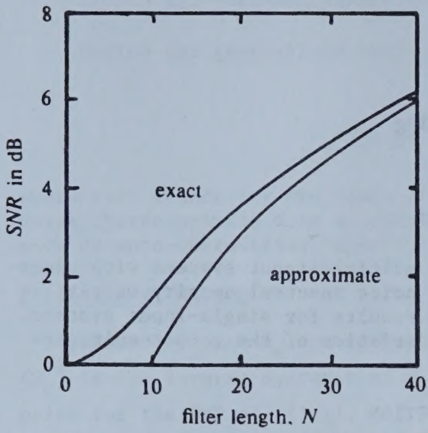


Fig. 9. Approximate and exact lower bound for Gaussian correlation.

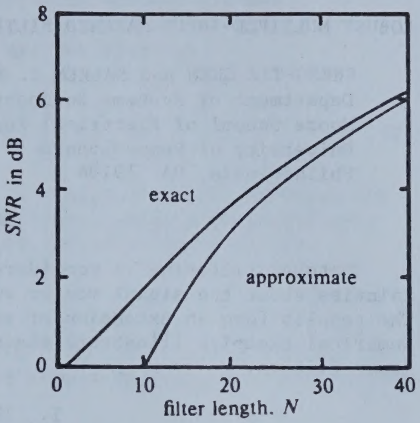


Fig. 10. Approximate and exact lower bound for hyperbolic secant correlation.

ROBUST MULTIPLE-INPUT MATCHED FILTERS [†]

CHENG-TIE CHEN and SALEEM A. KASSAM
Department of Systems Engineering
Moore School of Electrical Engineering
University of Pennsylvania
Philadelphia, PA 19104

ABSTRACT

Matched filtering is considered for multiple-input systems with uncertainties about the signal vector and the noise spectral density matrix. The results form an extension of earlier results for single-input systems. Numerical examples illustrate the characteristics of the robust multiple-input filters.

I. INTRODUCTION

For an M-input, L-output system the M x L matched filter matrix associates with each input-output pair a single-input, single-output (SISO) filter. There are various ways of defining output signal-to-noise ratio (SNR) for a multiple-input, multiple-output (MISO) system. Three useful definitions are given in [1]. It can be shown [1,2] that for these definitions, finding the matched filter matrix for MISO systems is essentially the same problem as that of finding the matched filter vector for multiple-input, single-output (MISO) systems. In this paper, we thus focus our attention exclusively on the MISO system.

The theory of matched filters which maximize the output SNR when signal and noise are exactly specified is well-known. However, due to a lack of complete knowledge, the signal and the noise specifications could be subject to a degree of uncertainty. Kuznetsov [3], Kassam et al. [4], and Poor [5] have considered this kind of problem by using the robustness concept which guarantees that the system performance will not be worse than some optimum lower bound. More specifically, let ρ denote the output SNR which, in general, is a functional of signal $\underline{S}$, noise characteristic N and filter $\underline{H}$, and let C_s, C_n denote the sets of allowable signals and noise characteristics, respectively. The idea then is to find the robust filter $\underline{H}_R$ and the least favorable signal-noise pair $(S_R, N_R) \in C_s \times C_n$ which together yield a saddle-point for the SNR, so that

$$\rho(\underline{H}_R, \underline{S}, N) \geq \rho(\underline{H}_R, \underline{S}_R, N_R) \geq \rho(\underline{H}, \underline{S}_R, N_R) \qquad (1)$$

for any $\underline{S} \in C_s$, $N \in C_n$ and any allowable $\underline{H}$. Although a general approach to robust matched filtering in a Hilbert space setting has been studied [5], all the cases considered are SISO systems. It is our purpose to investigate the theory of robust matched filters for MISO systems.

In section II, we first state a preliminary theorem from [5]. By using this preliminary theorem, a theorem for the robust MISO matched filters will be proved. Numerical examples will be presented to illustrate the application of these ideas in section III.

[†] This research is supported by the Office of Naval Research under Contract N00014-80-K-0945.

II. GENERAL RESULT FOR ROBUST MISO MATCHED FILTERS

Define the generalized SNR[5] in a Hilbert space as

$$\rho(\underline{H}, \underline{S}, N) = \frac{|<\underline{H}, \underline{S}>|^2}{<\underline{H}, N\underline{H}>} \qquad (2)$$

where <,> stands for the inner product in a Hilbert space and where the noise characteristic N is an operator involving second order statistics such as auto-correlation, spectral density, or co-variance matrix.

Preliminary Theorem

Suppose C_s and C_n are convex sets, $\underline{S}_R \in C_s$, $N_R \in C_n$, and $\underline{H}_R = N_R^{-1}\underline{S}_R$ (N_R^{-1} is the inverse operator of N_R). The triple $(\underline{H}_R, \underline{S}_R, N_R)$ is a saddle-point for the SNR of (2) if, and only if

$$2\mathrm{Re}\{<\underline{S}, \underline{H}_R>\} - <\underline{H}_R, N\underline{H}_R> \ \geq \ <\underline{S}_R, \underline{H}_R> \qquad (3)$$

for all $\underline{S} \in C_s$ and $N \in C_n$. A detailed proof was given in [5].

Consider the system model in Figure 1. This is a useful model for practical systems such as for array processing and multi-channel communication.

Let $\underline{S}(\omega) = [S_1(\omega), S_2(\omega), \ldots, S_M(\omega)]^T$ be the Fourier transform of the deterministic signal vector and let $\underline{G}(\omega) = [G_1(\omega), G_2(\omega), \ldots, G_M(\omega)]^T$ be the filter frequency response vector. With the assumption of wide-sense stationarity, we define $N(\omega)$ as the noise spectral density matrix whose ij-th element is the Fourier transform of the expectation of $n_i(t+\tau)n_j^*(t)$ where "*" means complex conjugate. Obviously, $N(\omega)$ is a Hermitian matrix whose eigenvalues are always non-negative.

With $\underline{H}(\omega) = [H_1(\omega), H_2(\omega), \ldots, H_M(\omega)]^T = e^{j\omega T}\underline{G}^*(\omega)$, it is easy to show that the output SNR at time T is

$$\rho(\underline{H}, \underline{S}, N) = \frac{\left|\frac{1}{2\pi}\int \underline{H}^\dagger(\omega)\underline{S}(\omega)d\omega\right|^2}{\frac{1}{2\pi}\int \underline{H}^\dagger(\omega)N(\omega)\underline{H}(\omega)d\omega} \qquad (4)$$

where "†" means conjugate transpose.

Consider the case where C_n is the set consisting of all noise spectral density matrices whose eigenvalues are bounded by known constraints, and satisfying r disjoint generalized power constraints. Specifically, with $N(\omega)$ expressed as $N(\omega) = p(\omega)\Lambda(\omega)p^\dagger(\omega)$ where $p(\omega)$ is an M x M unitary matrix and $\Lambda(\omega)$ is a diagonal matrix with non-negative i-th diagonal term $\lambda_i(\omega)$, the class C_n contains all matrices $N(\omega)$ satisfying (a) and (b):

(a) $\lambda_{Li}(\omega) \leq \lambda_i(\omega) \leq \lambda_{Ui}(\omega)$

where λ_{Li} and λ_{Ui} are given non-negative functions

(b) $$\sum_{i=k_j+1}^{k_{j+1}} \int \lambda_i(\omega)d\omega = 2\pi Q_j , \quad j = 1,2,\ldots,r \tag{5}$$

where $0 = k_1 < k_2 <\ldots< k_r < k_{r+1} = M$ and the Q_j are given.

We assume, of course, that (a) and (b) are not inconsistent. It is also assumed that $p(\omega)$ is known. For the case where $p(\omega)$ is a constant matrix P, the above class could model noise processes which are linear combinations of uncorrelated noise processes with spectral density matrix Λ, with transformation matrix P. Note that for the case r = 1, (b) becomes a constraint on sum of noise powers at the M inputs.

Let C_s be the set consisting of all signals $\underline{S}(\omega)$ which are in the neighborhood of a known nominal signal $S_0(\omega) = [S_{o1}(\omega), S_{o2}(\omega),\ldots,S_{oM}(\omega)]^T$ defined explicitly by

$$C_s = \{\underline{S}(\omega)| \sum_{i=m_j+1}^{m_{j+1}} \int |X_i(\omega) - X_{oi}(\omega)|^2 d\omega \leq 2\pi\delta_j, \; j = 1,2,..,q\} \tag{6}$$

where

$$0 = m_1 < m_2 <,\ldots, < m_q < m_{q+1} = M$$

$$\underline{X}(\omega) = [X_1(\omega), X_2(\omega),\ldots,X_M(\omega)]^T = p^{\dagger}(\omega)\underline{S}(\omega)$$

and

$$\underline{X}_o(\omega) = [X_{o1}(\omega), X_{o2}(\omega),\ldots,X_{oM}(\omega)]^T = p^{\dagger}(\omega)\underline{S}_o(\omega).$$

Note that for q = 1, the condition in (6) is equivalent to $\sum_{i=1}^{M} \int |S_i(\omega) - S_{oi}(\omega)|^2 d\omega \leq 2\pi\delta$.

Theorem

For noise and signal uncertainty classes (5) and (6), respectively, the the following least-favorable signal, noise pair $(\underline{S}_R(\omega), N_R(\omega))$ and the corresponding optimum filter

$$\underline{H}_R(\omega) = N_R^{-1}(\omega)\underline{S}_R(\omega) \tag{7}$$

form a saddle-point for SNR of (4):

$$\underline{S}_R(\omega) = p(\omega)\underline{X}_R(\omega) \tag{8}$$

and

$$N_R(\omega) = p(\omega)\Lambda_R(\omega)p^\dagger(\omega) \tag{9}$$

where

$$X_{Ri}(\omega) = \frac{\lambda_{Ri}(\omega)X_{oi}(\omega)}{\lambda_{Ri}(\omega) + \sigma_{sj}} \quad \text{for } m_j < i \le m_{j+1},\ j = 1,2,..,q \tag{10}$$

and $\Lambda_R(\omega)$ is diagonal with i-th diagonal term given by

$$\lambda_{Ri}(\omega) = \text{second largest of } \{\lambda_{Ui}(\omega),\ \lambda_{Li}(\omega),\ \sigma_{nj}|X_{Ri}(\omega)|\}$$

$$= \begin{cases} \lambda_{Ui}(\omega) & ,\ \text{for } \omega\varepsilon R_{Ui} \\ \lambda_{Li}(\omega) & ,\ \text{for } \omega\varepsilon R_{Li} \\ \sigma_{nj}|X_{Ri}(\omega)| & ,\ \text{for } \omega\varepsilon R_{Mi} \end{cases} \quad ,\ \text{for } k_j < i \le k_{j+1}\ , \tag{11}$$

if there exist non-negative σ_{sj}, $j = 1,2,...,q$ and σ_{nj}, $j = 1,2,...,r$ satisfying

$$\sum_{i=m_j+1}^{m_{j+1}} \frac{\sigma_{sj}^2}{2\pi} \int \frac{|X_{oi}(\omega)|^2}{(\lambda_{Ri}(\omega) + \sigma_{sj})^2}\, d\omega = \delta_j, \quad j = 1,2,...,q \tag{12}$$

and

$$\sum_{i=k_j+1}^{k_{j+1}} \{\int_{R_{Ui}} \lambda_{Ui}(\omega)d\omega + \int_{R_{Li}} \lambda_{Li}(\omega)d\omega + \int_{R_{Mi}} \sigma_{nj}|X_{Ri}(\omega)|d\omega\} = 2\pi Q_j, \quad j = 1,2,...,r \tag{13}$$

Proof

From the preliminary theorem we have to show that

$$2\mathrm{Re}\{\frac{1}{2\pi}\int \underline{H}_R^\dagger(\omega)\underline{S}(\omega)d\omega\} - \frac{1}{2\pi}\int \underline{H}_R^\dagger(\omega)N(\omega)\underline{H}_R(\omega) \ge \frac{1}{2\pi}\int \underline{H}_R^\dagger(\omega)S_R(\omega)d\omega \tag{14}$$

for all $\underline{S}(\omega)\varepsilon C_s$ and $N(\omega)\varepsilon C_n$. If we prove

$$\mathrm{Re}\{\frac{1}{2\pi}\int \underline{H}_R^\dagger(\omega)\underline{S}(\omega)d\omega\} - \frac{1}{2\pi}\int \underline{H}_R^\dagger(\omega)\underline{S}_R(\omega)d\omega \ge 0 \tag{15}$$

and

$$\frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)\underline{S}_R(\omega)d\omega - \frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)N(\omega)\underline{H}_R(\omega)d\omega \geq 0 \quad , \tag{16}$$

then (14) follows.

We first prove (15) is true for all $\underline{S}(\omega)\varepsilon C_s$. Substituting (7) and (8) into the left-hand side of (15), using the eigenexpansion for $N_R(\omega)$ and (10), and applying the Schwartz inequality, we can get

$$\begin{aligned}
&\mathrm{Re}\{\frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)\underline{S}(\omega)d\omega\} \\
&\qquad - \frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)\underline{S}_R(\omega)d\omega \geq -\sum_{j=1}^{q}\Big(\sum_{i=m_j+1}^{m_{j+1}}\frac{1}{2\pi}\int\frac{|X_{oi}(\omega)|^2}{(\lambda_{Ri}(\omega)+\sigma_{sj})^2}d\omega\Big)^{\frac{1}{2}}\cdot \\
&\qquad \Big(\sum_{i=m_j+1}^{m_{j+1}}\frac{1}{2\pi}\int|X_i(\omega)-X_{oi}(\omega)|^2 d\omega\Big)^{\frac{1}{2}} + \\
&\qquad \sum_{j=1}^{q}\sum_{i=m_j+1}^{m_{j+1}}\frac{\sigma_{sj}}{2\pi}\int\frac{|X_{oi}(\omega)|^2}{(\lambda_{Ri}(\omega)+\sigma_{sj})^2}d\omega \geq 0 \quad .
\end{aligned}$$

The last inequality follows by using (13) and the constraint in C_s.

We now prove (16) is true for all $N(\omega)\varepsilon C_n$. By using the eigenexpansion for the noise spectral density matrix and substituting (7), (8), and (9) into the left-hand side of (16), we can get

$$\begin{aligned}
&\frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)\underline{S}_R(\omega)d\omega - \frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)N(\omega)\underline{H}_R(\omega)d\omega \\
&= \sum_{i=1}^{M}\frac{1}{2\pi}\int\frac{|X_{Ri}(\omega)|^2}{\lambda_{Ri}^2(\omega)}(\lambda_{Ri}(\omega)-\lambda_i(\omega))d\omega
\end{aligned} \tag{17}$$

By separating the integral into three regions R_{Ui}, R_{Li}, and R_{Mi}, and noting that for $k_j < i \leq k_{j+1}$, $\frac{|X_{Ri}(\omega)|^2}{\lambda_{Ri}^2(\omega)} > \frac{1}{\sigma_{nj}^2}$ in R_{Ui}, $\frac{|X_{Ri}(\omega)|}{\lambda_{Ri}^2(\omega)} < \frac{1}{\sigma_{nj}^2}$ in R_{Li} , and $\frac{|X_{Ri}(\omega)|^2}{\lambda_{Ri}^2(\omega)} = \frac{1}{\sigma_{nj}^2}$ in R_{Mi}, (17) becomes

$$\frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)\underline{S}_R(\omega)d\omega - \frac{1}{2\pi}\int \underline{H}_R^{\dagger}(\omega)N(\omega)\underline{H}_R(\omega)d\omega$$

$$\geq \sum_{j=1}^{r} \sum_{i=k_j+1}^{k_{j+1}} \frac{1}{2\pi\sigma_{nj}^2}\{\int_{R_{Ui}}(\lambda_{Ui}(\omega) - \lambda_i(\omega)d\omega + \int_{R_{Li}}(\lambda_{Li}(\omega) - \lambda_i(\omega))d\omega$$

$$+ \int_{R_{Mi}}(\lambda_{Ri}(\omega) - \lambda_i(\omega))d\omega\}$$

$$= 0.$$

The last equality follows by using the condition (b) in C_n. Note that (13) and (14) put $\underline{S}_R(\omega)$ and $N_R(\omega)$ in C_s and C_n, respectively, and the proof is complete.

Comments

From the proof, we can see that the eigenexpansion reduces the problem to one equivalent to the uncorrelated noise case. In this case, the signals are also changed into signals related to the eigenspace of the noise spectral density matrix. Note (10) is similar to the result for single input cases, the original signals $S_i(\omega)$ having been changed into other signals $X_i(\omega)$, with the noise characterized by the eigenvalues.

For SISO systems, the robust matched filtering is trivial if the noise is white. However, it can be seen from the above results that for MISO systems, even if the known noise at each input is white (with different levels) and uncorrelated with each other, the SNR and robust solution will be influenced.

Note that only when the signals at each input are independently constrained and the noise is uncorrelated and independently constrained at each input will the earlier SISO results be directly applicable to the MISO problem. The signal model C_s of (6) is a generalization of the earlier SISO signal L_2-neighborhood model. The noise model C_n is also a generalization of the earlier "band-model" for the SISO system, when the noise is uncorrelated. The model when correlation is present is somewhat restricted by the requirement that $p(\omega)$ be known. However, it may be possible to embed special cases inside this model and use the results as a conservative solution.

Another useful point is that the problem formulation here is applicable to the one-input, discrete-time robust matched filtering problem [6]. If we change the expressions in the frequency domain into the corresponding ones in the discrete-time domain, the conclusions of the above generalized theorem can be drawn. In this case, M is the number of the time-sampling points of the input. Stationarity is not required for the noise covariance matrix. Letting q = M reduces the constraint in C_s to the L_∞-

neighborhood of a nominal signal, and the same results as those in [6] can be obtained for the known and uncorrelated noise. In fact correlated noise can be considered just as easily. These results are contained in [7].

III. EXAMPLES

For both examples in this section, we assume that the noise spectral density matrix is exactly known.

Example 1. Uncorrelated white noise

Consider the noise at the inputs to be uncorrelated and white. Let N_i be the noise power density at the i-th input. We set only one power constraint, i.e., q = 1. The unknown variable σ_s can be found from (12), which reduces to

$$\sum_{i=1}^{M} \frac{\sigma_s^2 E_{oi}}{(\sigma_s + N_i)^2} = \delta \tag{18}$$

where

$$E_{oi} = \frac{1}{2\pi} \int |S_{oi}(\omega)|^2 d\omega \; .$$

The least-favorable signals and the robust matched filters are then obtained from (10) and (7).

The results in Table I are based on the following data: M = 5, $E_{o1} = 4.0$, $E_{o2} = 0.98$, $E_{o3} = 2.5$, $E_{o4} = 1.0$, $E_{o5} = 0.9$, $N_1 = 0.45$, $N_2 = 4.0$, $N_3 = 1.0$, $N_4 = 2.5$, $N_5 = 0.9$.

It can be seen from Table I that even if the inputs are nominal signals, using the robust filters do not degrade the SNR too much. The above results show that the robust solution and SNR for uncorrelated, white noise are not trivial for MISO systems.

Example 2

In this example, we will examine the SNR behavior as a function of time delay for a 2-element sonar array system in Figure 2. The time delay τ in the second channel should be $\tau = d\sin\theta/C$, where C = velocity of sound in water.

Let us take

$$N(\omega) = \begin{bmatrix} 2 & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & 1 \end{bmatrix} N_o(\omega) \quad \text{where} \quad N_o(\omega) = \frac{2E_n a}{\omega^2 + a^2}$$

and $C_s = \{\underline{S}(\omega) \mid \sum_{i=1}^{2} \frac{1}{2\pi} \int |S_i(\omega) - S_{oi}(\omega)|^2 \, d\omega \leq \delta\}$ where the nominal signals are

$$S_{o1}(\omega) = S_{o2}(\omega) = S_{LP}(\omega) = \begin{cases} \left(\frac{\pi E_s}{W}\right)^{\frac{1}{2}} & , \text{ for } |\omega| \leq W \\ 0 & , \text{ otherwise} \end{cases}$$

with $\delta = 3.89$, $a^2 = 200$, $E_n = 1.2$, $E_s = 2.5$, and $W = 20\pi$. We can solve σ_s from (12), which turns out to be 0.76. The robust matched filter $\underline{H}_R(\omega)$ is then obtained from (7).

The curves in Figure 3 are the SNR in db vs. t_o when $S_1(\omega) = S_{LP}(\omega)$ and $S_2(\omega) = S_{LP}(\omega)e^{-j\omega t_o}$ for curves 1 and 2, and when $S_1(\omega) = S_{R1}(\omega)$ and $S_2(\omega) = S_{R2}(\omega)e^{-j\omega t_o}$ for curves 3 and 4. The filter for curves 1 and 3 is the nominal filter $\underline{H}_o(\omega) = N^{-1}(\omega)\underline{S}_o(\omega)$, and that for curves 2 and 4 is the robust matched filter $\underline{H}_R(\omega)$. It can be seen from the figure that the SNR curves with $\underline{H}_R(\omega)$ are smoothed out, as expected. Although the mainlobe of SNR curves with $\underline{H}_o(\omega)$ is narrower, the sidelobe behavior is worse when we compare curve 1 with curve 2. In this example, in order to clearly observe SNR characteristics, we used an unusually high signal uncertainty. In the case of small δ, the SNR performance with $\underline{H}_R(\omega)$ will be better.

IV. CONCLUSION

In this paper we extended previous results on robust matched filtering to the multiple-input case. A theorem was proved for signals modeled as lying in a generalized L_2-neighborhood of a nominal signal, with a noise spectral matrix class where the eigenvalues are bounded and satisfy a generalized power constraint.

It was shown that the results are directly applicable to one-input, discrete-time robust matched filtering.

We did not put any constraint on the eigenvectors of the noise spectral matrix. This is an interesting research topic in this area of robust MISO matched filters.

REFERENCES

1. A. V. Balakrishnan, "Matched Filters for Multiple Processes", IRE Trans. Information Theory, Vol. IT-7, p.52, Jan. 1961.

2. A. V. Balakrishnan, "Estimation and Detection Theory for Multiple Stochastic Processes", J. Math. Analysis and Application, Vol. 1, pp. 386-410, December 1960.

3. V. P. Kuznetsov, "Stable detection when the signal and spectrum of normal noise are inaccurately known", Telecommun. Radio Eng., Vol. 30/31, pp. 58-64, March 1976.

4. S. A. Kassam, T. L. Lim, and L. J. Cimini, "Two dimensional Filters for Signal Processing under Modelling Uncertainties", IEEE Trans. Geosci. and Remote Sensing, Vol. GE-18, pp. 331-336, Oct. 1980.

5. H. V. Poor, "A general Approach to Robust Matched Filtering", Proc. 22nd Midwest Symp. Circuits and Systems, pp. 514-518, June 1979.

6. S. Verdu and H. V. Poor, "Least Favorable Signals and Noise for Discrete Time Robust Robust Matched Filtering", to be published.

7. C. T. Chen and S. A. Kassam, "General results on multiple-input robust matched filters", to be published.

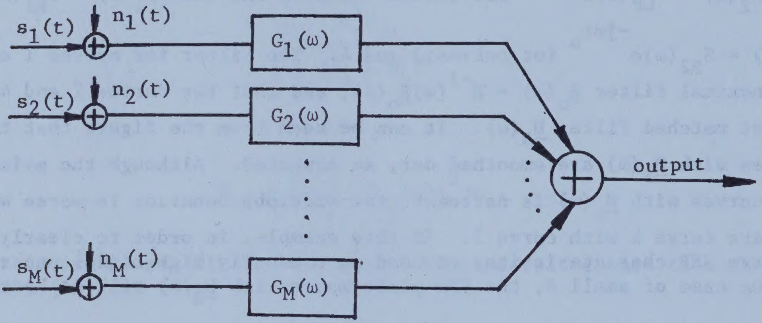


Figure 1. MISO system

Table 1. Numerical results for Example 1.

δ	σ_s	$\rho(\underline{H}_R, \underline{S}_R, N)$	$\rho(\underline{H}_R, \underline{S}_o, N)$	$\rho(N^{-1}\underline{S}_o, \underline{S}_R, N)$
0	0.00	13.03	13.03	13.03
1	0.34	5.29	12.81	5.17
2	0.65	3.18	12.35	3.01
3	1.05	1.97	11.83	1.79
4	1.61	1.19	11.35	1.04
5	2.47	0.69	10.71	0.56

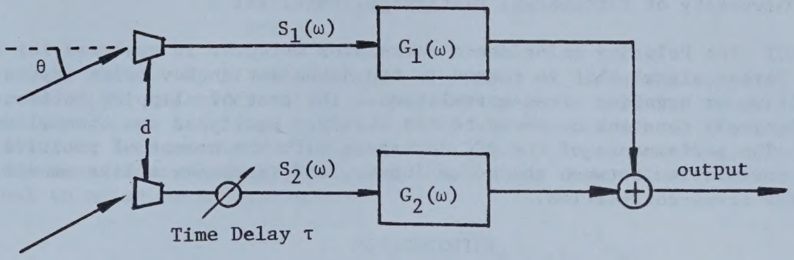

θ
d
$S_1(\omega)$
$G_1(\omega)$
$S_2(\omega)$
$G_2(\omega)$
Time Delay τ
output

Figure 2. System for Example 2.

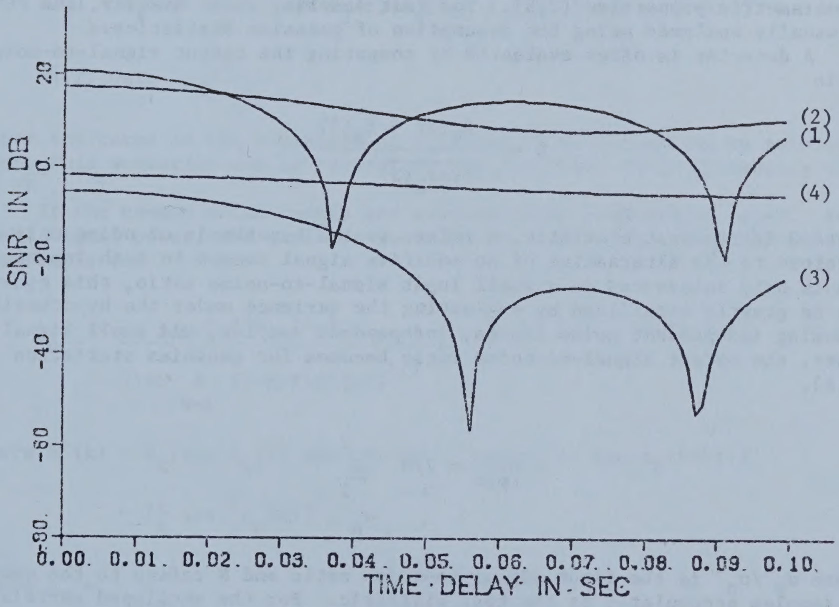


Figure 3. SNR curves for Example 2.

"A COMPARISON OF PALARITY COINCIDENCE CORRELATION WITH UNCLIPPED DETECTORS FOR CORRELATED NOISE INPUTS"

M. Kanefsky and A. Q. K. Rajput
University of Pittsburgh, Pittsburgh, PA 15261

ABSTRACT The Polarity coincidence correlator detector is analyzed for a Small Markov signal that is common to two dependent Markov noise inputs with a positive or negative cross-correlation. The cost of clipping is found to be relatively constant compared to the standard unclipped two channel detectors. The performance of the PCC decreases with the amount of positive cross correlation between the noise inputs, and increases a like amount for negative cross-correlation.

INTRODUCTION

Polarity coincidence correlation is a well-documented technique for detecting a random signal imbedded in additive noise [1,2,3,4,5]. Its primary applications have involved acoustic signals such as those arising in passive sonar detection [1,4] and seismic detection [6]. It is commonly assumed, when dealing with low-frequency acoustic signals, that the two noise inputs to the PCC detector are statistically independent. Under this assumption the dectector is very efficient [1,3,6]. When the signal and noise samples are all statistically independent, the PCC has interesting nonparametric properties [2,5]. For fast sampling rates however, the PCC is usually analyzed using the assumption of guassian statistics.

A detector is often evaluated by computing the output signal-to-noise ratio

$$D = \frac{E_K(S) - E_H(S)}{\sqrt{\mathrm{Var}_K(S)}} \qquad (1)$$

where S is the test statistic, H refers to the hypothesis of noise only and K refers to the alternative of an additive signal common to both inputs. If one is only interested in a small input signal-to-noise ratio, this equation can be greatly simplified by evaluating the variance under the hypothesis. Assuming independent noise inputs, independent samples, and small signal power, the output signal-to-noise ratio becomes for gaussian statistics [1,2],

$$D_{pcc} = \frac{2\sqrt{N}}{\pi} \frac{\sigma_s^2}{\sigma_n^2} \qquad (2)$$

where σ_s^2/σ_n^2 is the input signal-to-noise ratio and N refers to the number of samples accumulated by the test statistic. For the unclipped correlator, the output signal-to-noise ratio is $\sigma_s^2/\sigma_n^2 \sqrt{N}$ [1,2] and the cost of clipping is $\pi/2$ or 2db. The optimum detector, given that one knows or can measure the noise variance computes the square of the sum of the inputs and has a signal to noise ratio $\sqrt{2}$ times larger than the unclipped correlator. However, if this detector variance can only be estimated in the presence of the signal, its performance is slightly worse than that of the unclipped correlator which, under these conditions, becomes optimum. For these reasons we will compare the PCC to both analog detectors.

If one wishes to change some of the assumptions, it is necessary to examine the test statistic more closely.

$$S = \sum_{k=1}^{N} \operatorname{sgn} x_1(t+k\tau) \operatorname{sgn} x_2 (t+k\tau) \qquad (3)$$

where sgn $x(t) = 1$, if $x(t) > 0$ and -1 if $x(t) < 0$, τ is the sampling interval, and $x_1(t)$ and $x_2(t)$ refers to the two inputs. From Van Vleck [8] it is known that $E[\operatorname{sgn} x_1(t) \operatorname{sgn} x_2(t)] = (2/\Pi) \sin^{-1} \rho$ where ρ is the correlation coefficient between the two inputs. If ρ_H and ρ_K represent the correlation coefficient under the hypothesis and alternative respectively, the output signal to noise ratio becomes:

$$D_{pcc} = \frac{(2/\pi)\sqrt{N}\,[\sin^{-1}\rho_K - \sin^{-1}\rho_H]}{[1-\{(2/\pi)\sin^{-1}\rho_K\}^2 + 2\sum_{k=1}^{N}[1-k/N][R(k)-\{(2/\pi)\sin^{-1}\rho_K\}^2]]^{1/2}} \qquad (4)$$

where, $R(k) = E_K\,[\operatorname{sgn} x_1(t) \operatorname{sgn} x_2(t) \operatorname{sgn} x_1(t+k\tau) \operatorname{sgn} x_2(t+k\tau)]$. (5)

Since $x_i(t)$ and $x_i(t+k\tau)$ become independent as k increases, it follows that

$$R(k) \underset{k\to\infty}{\simeq} E_K[\operatorname{sgn} x_1(t) \operatorname{sgn} x_2(t)]E_K[\operatorname{sgn} x_1(t+k\tau) \operatorname{sgn} x_2(t+k\tau)] \qquad (6)$$

$$= [(2/\pi)\sin^{-1}\rho_K]^2.$$

Hence the terms in the summation of equation(4)are decreasing to zero and since this summation can be truncated, the detection index ultimately varies as $\sqrt{N}$.

If the common noise inputs are assumed to be independent, $\rho_H = 0$. For small signal power, the variance is evaluated under the assumption of the hypothesis and

$$D_{pcc} = \frac{2/\pi\sqrt{N}\,\sin^{-1}[\rho_K]}{[1+2\sum_{k=1}^{N}(1-k/N)R'(k)]^{1/2}}, \qquad (7)$$

where $R'(k) \simeq E_H[\operatorname{sgn} x_1(t) \operatorname{sgn} x_1(t+k\tau) \quad \operatorname{sgn} x_2(t) \operatorname{sgn} x_2(t+k\tau)]$

$$\simeq [\tfrac{2}{\pi}\sin^{-1}\rho_n(k)]^2,$$

where $\rho_K = \dfrac{\sigma_s^2}{\sigma_n^2 + \sigma_s^2}$, $\quad \rho_n(k) = \dfrac{R_n[k\tau]}{\sigma_n^2}$,

and $R_n(k\tau)$ is the autocorrelation of the noise. For independent samples $[\rho_n(k) = 0]$, equation(7)reduces to that of equation(2). Equation(7)has been evaluated for dependent samples [3] where the cost of clipping has been found to decrease as the sampling rate increases.

For dependent noise inputs, we must compute R(k) which involves 16 integrals of the type

$$\int_0^\infty \int_0^\infty \int_0^\infty \int_0^\infty (2\pi)^{-2} [\det P]^{-1/2} \exp[-1/2\ \underline{x}\ P^{-1} \underline{x}']\ d\underline{x} \quad , \qquad (8)$$

Where $\underline{x}$ is the Vector (x_1,x_2,x_3,x_4) and the covariance matrix P is evaluated for each of the 16 integrals for each value of k.

A general solution is available [7] for the computation of R(k). This paper utilizes this result for computing the performance of the PCC for a positive and negative cross-correlation of the two noise inputs, with a weak common signal.

A cross-correlation function whose shape is the same as the autocorrelation will be assumed and the relationship between the cost of clipping and the degree of correlation will be examined. The cost comparisons will be made with the correlator and optimum (sum and square) detector, for both positive and negative cross-correlations. The sum and square detector is optimum only under the assumption of independent inputs in addition to the requirement of a known or measurable noise variance. For convenience we will continue to refer to this detector as the optimum detector even though it clearly is not for correlated noise inputs.

COMPUTATION OF D_{pcc}

The expected value of the product of four clipped gaussian inputs is given by [7] as

$$R[k] = \pi^{-2} [\det P]^{-1/2} C_{11}^{-2} \sum_{\substack{i=0 \\ \text{odd}}}^{\infty} \sum_{j=0}^{\infty} \sum_{r=0}^{\infty} \sum_{\substack{y=j \\ \text{odd}}}^{i+j} \sum_{s=0}^{j} \sum_{\substack{x=r+s \\ \text{odd}}}^{r+s+i+j-y} \alpha(i,j,r,y,s,x)\ \beta(i,j,r,y,s,x), \qquad (9)$$

$$\text{where } \alpha(i,j,r,y,s,x) = \frac{2^{i+j+r} \left(\frac{i-1}{2}\right)! \left(\frac{y-1}{2}\right)! \left(\frac{i-x-y+2y+2r-1}{2}\right)!}{(j-s)!\ (i+j+s+\gamma-x-y)!\ (y-j)!\ S!\ (x-s-r\ !\ r!} ,$$

$$\text{and } \beta(i,j,r,y,s,x) = C_{12}^{\,y-j} \cdot C_{34}^{\,r} \cdot C_{23}^{\,s} \cdot C_{13}^{\,x-s-r} , C_{24}^{\,j-s} \cdot C_{14}^{\,i+j+s+r-x-y}$$

The coefficients C_{ij} are the normalized elements of $[P]^{-1}$ where both [P] and $[P]^{-1}$ have even symmetry.

The detection parameter for the PCC as given by equation (4) can be evaluated using a computer program to compute R(k) from equation (9). For a small signal strength we can approximate $\mathrm{Var}_K(S_D)$ by $\mathrm{Var}_H(S_D)$ and equation (4) by

$$D_{pcc} = \frac{(2/\pi)\ \sqrt{N}\ \{\sin^{-1}\rho_k - \sin^{-1}\rho_H\}}{\{1-(2/\pi\cdot\sin^{-1}\rho_H)^2 + 2\sum_{k=1}^{N} (1-k/n)[R_H(k) - (2/\pi\cdot\sin^{-1}\rho_H)^2]\}^{1/2}} \qquad (10)$$

For the Markov process, and most others, the correlation between input samples decreases as k incrases and for k large enough, $R_H(k)$ in equation (10) approaches $(2/\pi \sin^{-1}\rho_H)^2$. For the case considered in this paper, the summation can be truncated at k=30.

The infinite summation of equation 9 have to be truncated to some finite value T forcing a compromise between CPU time and truncation error. The truncation error is found to be more severe the more the inputs are correlated. The CPU time required on a DEC-10 computer to evaluate equation 9 is plotted in fig. 1 for various truncation values. We observe an increase in CPU time of roughly 16 if the truncation T is increased from 5 to 9. For small amounts of correlation, T=5 is adequate [7] and the total computation time for equation(10), where the summation is truncated at 30, is roughly 30 seconds.

There were no difficulties in computing $R_H(k)$ for cross-correlation coefficients in the range between 0 and 0.5. However, for those coefficients greater than 0.5 and for small values of k $(1 < k < 3)$, equation 9 does not converge as the truncation is increased. In this unstable region (which is evident from the eigen values of the P^{-1} matrix and hence can be isolated) computation approximation may be possible.

The computation can be made stable with rapid convergence if the elements in the diagonal of the P matrix are all increased sufficiently. If the results of this computation are compared with the true results for those values where convergence is no problem, approximations can often be made. Such a comparison is shown in fig. 2 for a cross-correlation coefficient of 0.6, where the approximation for three points where convergence did not take place are self-evident. The results in this paper employ these approximations for correlation coefficients greater than 0.5 or less than -0.5.

COMPUTATION OF DETECTION PARAMETERS FOR CORRELATOR AND OPTIMUM DETECTORS

The detection parameter, as defined in equation (1) can be written, for small signals, as

$$D = \frac{E_K[S_D] - E_H[S_D]}{\sqrt{\mathrm{Var}_H[S_D]}}, \tag{11}$$

where, for the correlator

$$S_c = \sum_{i=1}^{N} x_1(t_i)\, x_2(t_i), \tag{12}$$

where $x_i(t)$ refer to the two inputs.

For the purpose of this evaluation we will assume a first order Markov process or

$$R(k) = \sigma_n^2 e^{-\alpha_0 k}, \tag{13}$$

and a similar shaped cross-correlation or

$$R_{n_1 n_2}(k) = \alpha\, R(k), \tag{14}$$

where α varies between -1 and +1. With these assumptions, equation (11) can be evaluated for the correlator resulting in:

$$D_c = \frac{\sqrt{N}\ \sigma_s^2/\sigma_n^2}{\sqrt{(1+\alpha^2)\ [1+2\sum_{k=1}^{N} (1-k/N)\ e^{-2\alpha_0 k}]}} \ . \tag{15}$$

The output signal-to-noise ratio for the optimum detector, defined as

$$S_{opt} = \sum_{i=1}^{N} [x_1(t_i) + x_2(t_i)]^2 \tag{16}$$

can also be computed under the previous assumptions. This computation is long but straight forward and reduces to

$$D_{opt} = \frac{\sqrt{2N}\ \sigma_s^2/\sigma_n^2}{(1+\alpha)\sqrt{1+2\sum_{k=1}^{N} (1-k/N)\ e^{-2\alpha_0 k}}} \tag{17}$$

where α can be positive or negative.

COMPARISON OF THE PCC FOR DEPENDENT NOISE INPUTS WITH THE CORRELATOR AND OPTIMUM DETECTORS

The performance parameter (D_{pcc}) of the PCC is given by equation 10 where $R_H(k) = E_H\{\mathrm{sgn}\ n_1(t)\ \mathrm{sgn}\ n_2(t)\ \mathrm{sgn}\ n_1(t+k)\ \mathrm{sgn}\ n_2(t+k)\}$. (18)
We will assume $\rho_H = E[n_1(t) \cdot n_2(t)]/\sigma_n^2 = \alpha$,
where α ranges from -1 to 1. More specifically we will assume that the noise inputs and common signal are rapidly varying Markov processes $[R(k\tau)=\sigma^2 e^{-k/4}]$ and the cross-correlation function between them is $Rn_1n_1(k\tau)=\sigma_n^2 \alpha e^{-k/4}$. Since a Markov process has an auto-correlation function of $\sigma_n^2 e^{-2\pi\beta\tau}$ where β is the signal bandwidth, this corresponds to a sampling rate of $8\pi\beta$. This is sufficiently fast in that the output signal-to-noise can not be increased by faster sampling [3]. The analysis in no way depends on these assumptions rather they are made for convenience.

In equation (10), ρ_k becomes $\dfrac{\sigma_s^2/\sigma_n^2 + \alpha}{\sigma_s^2/\sigma_n^2 + 1}$.

Using Taylor's expansion it is seen that

$$\sin^{-1}\rho_k \simeq \sin^{-1}(\alpha) + \frac{(1-\alpha)}{\sqrt{(1-\alpha^2)}}\ \sigma_s^2/\sigma_n^2 \tag{19}$$

Therefore, the numerator of equation (10) becomes

$$\frac{2}{\pi}\ \sqrt{N}\sqrt{\frac{1-a}{1+a}}\ \sigma_s^2/\sigma_n^2 \tag{20}$$

The covariance matrix is given by

$$P(k) = \sigma_n^2 \begin{vmatrix} 1 & \alpha & e^{-k/4} & \alpha e^{-k/4} \\ \alpha & 1 & \alpha e^{-k/4} & e^{-k/4} \\ e^{-k/4} & \alpha e^{-k/4} & 1 & \alpha \\ \alpha e^{-k/4} & e^{-k/4} & \alpha & 1 \end{vmatrix} \tag{21}$$

Because the results are quite different for positive and negative cross-correlations they will be evaluated and plotted separately.

The detection parameter D_{pcc}, or output signal-to-noise ratio, is plotted in fig. 3 as a function of the amount of positive cross-correlation. Shown in the same figure are the performance of the unclipped correlator and the optimum detector. As expected, the performance of all three detectors fall off as the noise correlation increases. The cost of clipping in decibels is shown in fig. 4. It can be observed that while the performance of both the PCC and the optimum detector fall off sharply with an increase in correlation, the cost of clipping does not change substantially, increasing from 2 db to 3 db for the assumed sampling rate. The performance of the unclipped correlator does not fall off as sharply and, for a sufficiently high correlation coefficient, it out-performs the "optimum" detector.

All three detection parameter curves are repeated in fig. 5 for negative cross-correlation functions. The cost of clipping is hown in fig. 6. In this case, the performance of the PCC and the optimum detectors increase by roughly the same amount (i.e. the cost of clipping is between 2 and 3 db), The increase in performance is essentially the same amount for negatively correlated noise as the decrease for positively correlated noise. The unclipped correlator performs the same for negative as for positive cross-correlations and is, therefore, less attractive for negatively correlated noise inputs.

REFERENCES

1. P.M. Schultheiss, and F.B. Tuteur, "Optimum and suboptimum detection of directional gaussian signals in isotropic gaussian noise field", Part II: Degradation of detectability due to clipping", IEEE Trans. of Milit. Electr., Vol, MIL-9, pp. 208-211, July/October 1965.
2. S.S. Wolff, J.B. Thomas, and J.R. Williams, "The Polarity coincidence correlator; A nonparametric detection device", IRE Trans. on Info. Theory, Vol, IT-8, pp. 1-19, Jan. 1962.
3. M. Kanefsky and J.B. Thomas, "Polarity coincidence correlation using de pendent samples", Proc. 1964 Nat'l Electr. Cong., Vol. 20 p. 714.
4. M. Kanefsky, "Detection of weak signals with polarity coincidence arrays", IEEE Trans. on Inf. Theory, Vol. IT-12, No. 2, April 1966.
5. M. Kanefsky and J.B. Thomas, "On polarity detection schemes with non-gaussian inputs", Journ. of the Franklin Institute, Vol 280, No. 2, April 1965.
6. J. Capon, "Application of detection and estimation theory to large array seismology", Proc. IEEE, Vol 58, No. 5, pp. 760-770, May 1970.
7. A.Q.K. Rajput, M. Kanefsky, "On evaluating polarity coincidence correlation when the two inputs are statistically dependent", IEEE Trans. on Inf. Theory., (Submitted for publication).
8. J.H. Van Vleck, and D, Middleton, "The spectrum of clipped noise", Proc. Ieee Vol. 54, No. 1, Jan. 1966, pp. 2-19.

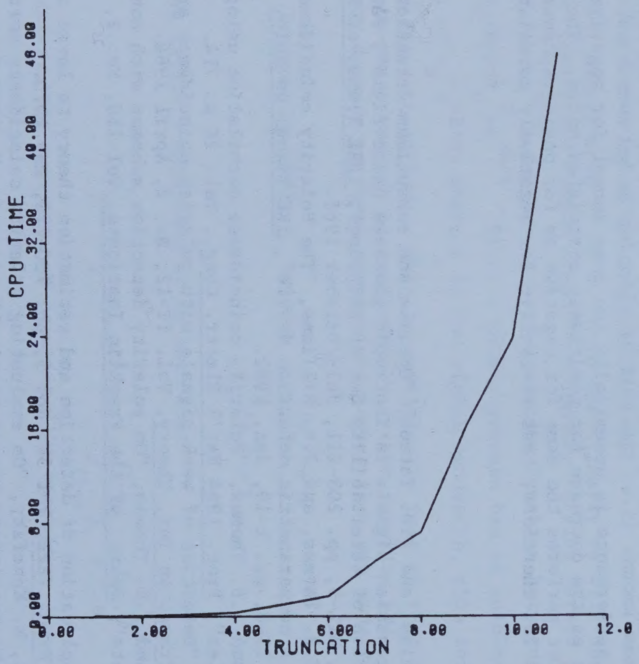


FIG.1 DEPENDENCE OF CPU TIME(SEC) VS TRUNCATION

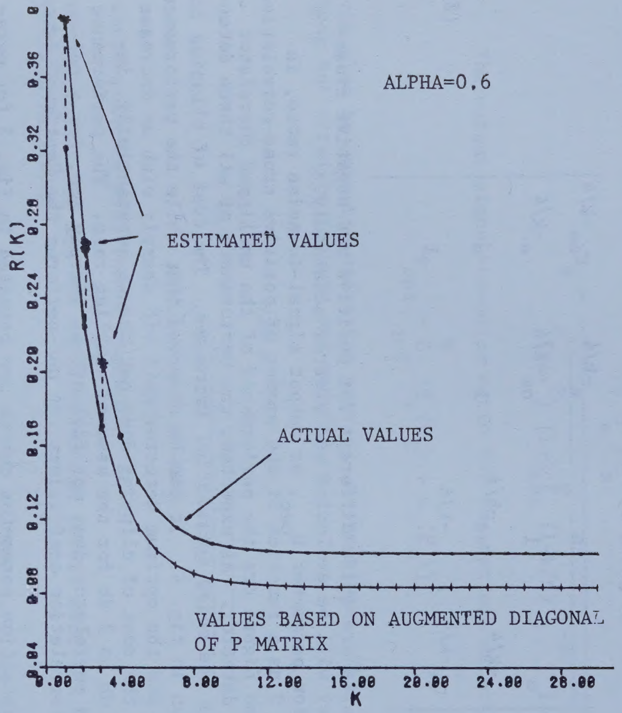


FIG.2 APPROXIMATION OF R(k)

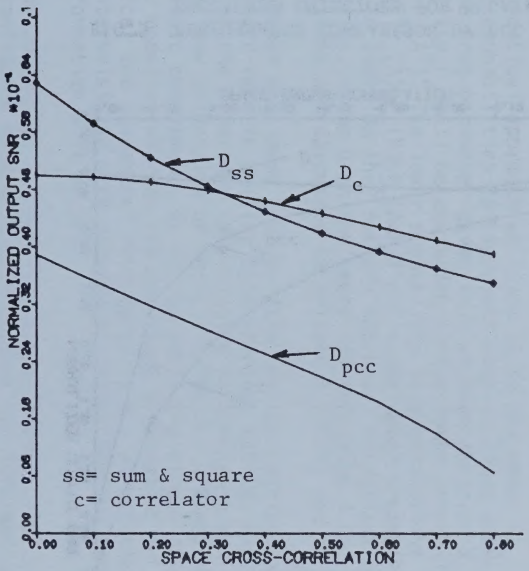


FIG.3 PERFORMANCE COMPARISON OF PCC WITH UNCLIPPED DETECTORS FOR POSITIVELY CORRELATED NOISE INPUTS

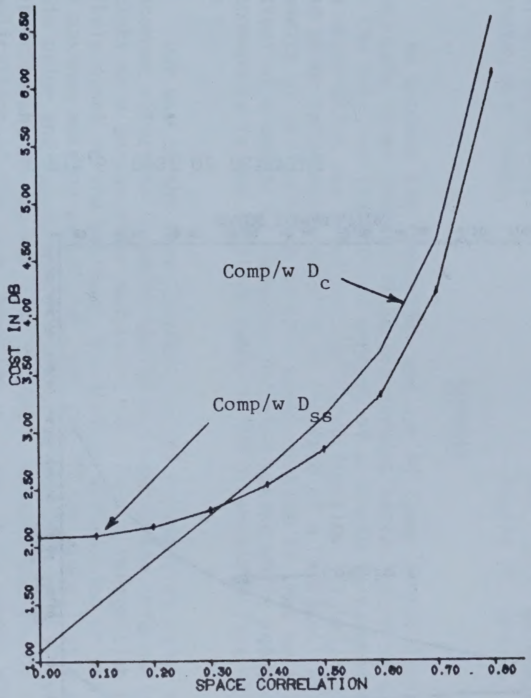


FIG.4 COST OF CLIPPING FOR POSITIVELY CORRELATED INPUTS

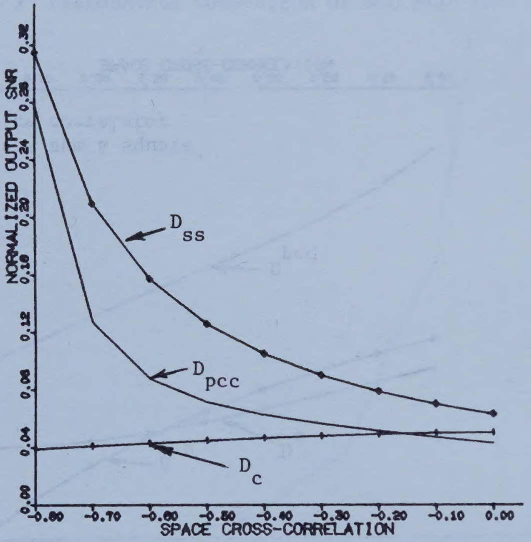


FIG.5 PERFORMANCE COMPARISON OF PCC WITH UNCLIPPED DETECTORS FOR NEGATIVELY CORRELATED INPUTS

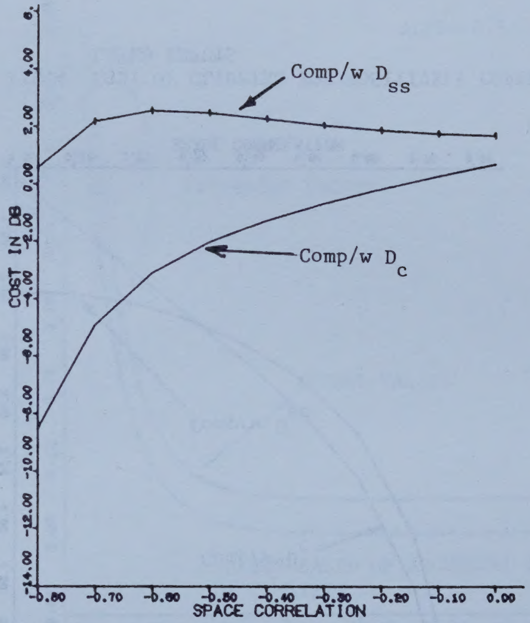


FIG.6 COST OF CLIPPING

DECISION MAKING WITH LIKELIHOOD BASED LOWER PROBABILITIES

by

Terrence L. Fine
School of Electrical Engineering
Cornell University
Ithaca, N.Y. 14853

Marco Wolfenson
Dept. of Electrical Systems
Fed. Univ. of Pernambuco
Recife PE, Brazil

SUMMARY

We explore the generation and use of lower probabilities (the conjugate upper probabilities are perhaps better known as capacities) in the familiar statistical context of a countable parameter set $\Theta=\{\theta\}$, an observation set $X=\{x\}$, a likelihood function $\{(f_\theta(x),x,\theta):x\varepsilon X,\ \theta\varepsilon\Theta\}$, and prior information about Θ in the form of a 'salient' prior π. The novelty of our approach to this basic problem is that we will preserve essential elements of Bayesian decision making while being able to accommodate to the decision maker's (DM's) possible lack of confidence in π.

Our use of the term 'salient prior' is meant to refer to the not uncommon situation in which the DM has some prior information that points to a unique prior measure π on Θ (e.g., recognition of equally likely cases on the basis of symmetrical information or use of some form of maximum entropy principle), yet the DM can place little confidence in this prior and has no alternative of comparable credibility. In such a situation the DM may not wish to commit herself too firmly to the implications of the usual Bayesian approach as it cannot differentiate between strongly and weakly held priors. Our approach replaces the usual posterior measure by an interval-valued posterior with the interval width reflecting the DM's lack of conviction about π.

The mathematical approach we adopt is to use upper $(\bar{P}_x(A))$ and lower $(\underline{P}_x(A))$ probabilities to represent what is known about $A\subset\Theta$ containing the true parameter given that we have observed x. The upper probability $\bar{P}_x$ will be conjugate to the lower probability $\underline{P}_x$, i.e., $\bar{P}_x(A)=1-\underline{P}_x(A^c)$. The lower probability is superadditive (A disjoint from B implies $\underline{P}_x(A)+\underline{P}_x(B)\leq\underline{P}_x(A\cup B)$) while the upper probability is subadditive. In addition they satisfy the usual normalization conditions of non-negativity, value 0 on the empty set and value 1 on Θ. Properties of these functions are discussed in Wolfenson [1979] and in Huber [1980].

We link upper and lower probabilities to the statistical setup by considering their role in utility type decision making. Our approach follows that of the Bayesians in focussing on dominance and coherence but introduces new principles of being a basis for a Bayes decision rule and of least commitment. We do not adopt the viewpoints of inference or scientific modeling wherein one processes information in accordance with theoretical and conceptual principles (e.g., asymptotic consistency, confidence intervals) and with only minor regard for the use to which the processed information will be put.

Our major conclusion is that if given a salient prior π in which we have little confidence then we should process our information so as to generate the least committal Bayes basis (LCBB) lower probability family of posteriors $\{\underline{P}_x\}$. Our major theorem is that the LCBB family is unique with the following simple analytical form:

$$\underline{P}_x(A) = \begin{cases} 0 \text{ if } \mu_x(a)<1/2 \\ 2\mu_x(A)-1 \text{ if } \mu_x(A)\geq 1/2 \end{cases} \qquad \bar{P}_x(A) = \begin{cases} 1 \text{ if } \mu_x(A)\geq 1/2 \\ 2\mu_x(A) \text{ if } \mu_x(A)<1/2, \end{cases}$$

where $\mu_x(A)$ is the usual posterior density

$$\mu_x(A) = \sum_A \pi(\theta)\, f_\theta(x) / \sum_\Theta \pi(\theta) f_\theta(x).$$

The DM's lack of confidence in π is reflected in the generally non-zero interval width $\bar{P}_x(A)-\underline{P}_x(A)$ attached to the parameter subset A given the observation $x \epsilon X$. The usual Bayesian posterior would of course be just μ_x.

Given $\underline{P}_x$, an action set $\mathcal{A} = \{a\}$, and a utility-valued function

$$v\colon \Theta \times \mathcal{A} \to R^1,$$

then one can calculate upper $\bar{E}_x V(\theta,a)$ and lower $\underline{E}_x V$ expectations for a given action a according to Dempster [1967]. Hence to each action a we correspond an interval $(\underline{E}_x, \bar{E}_x)$ of expected utilities. Two actions can now be compared on the basis of their associated utility intervals. A realistic consequence of our approach is the possibility that two actions may be incomparable. This incomparability can be resolved through the introduction of additional decision making principles. We will provide an example of such a principle in a variant of minimax risk based upon maximizing the lower expectation $\underline{E}_x V$.

The preceding ideas are discussed more fully in a paper by the authors that has been accepted for publication in the J. Amer. Stat. Assn.

References

A. Dempster [1967], Upper and lower probabilities induced by multivalued mapping, Ann. Math. Stat., 38, 325-339

P. Huber [1980], Robust Statistics, J. Wiley: New York.

M. Wolfenson [1979], Inference and Decision Making Based Upon Interval-Valued Probability, PhD. Dissertation, Cornell University, Ithaca, N.Y.

DETECTION OF CONSTANT SIGNALS IN SKEWED NOISE

Y.F. HUANG and J.B. THOMAS
Department of Electrical Engineering
and Computer Science
Princeton University
Princeton, N.J. 08544

ABSTRACT

The sensitivity of the linear detector to the underlying noise skewness is investigated. It is assumed that the skewness is small enough so that an approximate analysis with simpler expressions can be achieved. A modified linear detector is proposed which may be less sensitive to the noise skewness and can be applied to situations when the underlying noise is slightly non-Gaussian due to skewness. In addition, some examples of skewed density functions are discussed together with the normal density functions. The associated Neyman-Pearson optimal statistics are also examined.

I. Introduction:

The symmetry of the noise density function has been a common assumption in communication problems. In particular, most research on detection problems has been carried out with this assumption since it usually makes the problems more tractable and also provides simpler solutions. Typical examples appeared in the design of robust detectors [1],[2], and of nonparametric detectors, e.g. the generalized sign detectors proposed by Kassam and Thomas [3]. However, this symmetry assumption may not be exactly true in all practical situations. In such situations, the detection problem may still be solved for the Neyman-Pearson optimal detector or the locally optimal detector. However, there are now some additional difficulties associated with the implementation of either of these two detection schemes. The test statistics are usually complicated or intractable, and to implement either of these two schemes exactly requries a complete knowledge of the underlying noise density functions. Furthermore, the performance of the detector may be sensitive to the inaccuracy of the presumed statistics, and, in particular, to the exact amount of skewness.

When the skewness is not pronounced, the noise density function is often treated as symmetric. However, a more interesting problem is to study the sensitivity of the conventional optimal detectors, e.g. the linear detector, to noise skewness; that is, to relate the performance of these optimal detectors to some skewness measure. In addition, and more importantly, it may be desirable to implement a detection scheme which is less sensitive to the skewness of the noise density. These issues do not appear to have been discussed extensively in the literature, partly because the usual skewness measures [4][5] do not appear explicitly in the conventional test statistics. Johnson [6] discussed the effect of population skewness on the t-variable and proposed a procedure that reduces this effect on the distribution of the t-variable so that tests about the mean can be more correctly calulated. The method used by Johnson is to modify the t-variable for the non-normality of the population distribution using properties of data such that the adjusted version has a t-distribution to a sufficiently good approximation. The form of such a modified t-variable is derived by using the Cornish-Fisher expansion.

This paper discusses the effect of noise skewness on the performance of the linear detector. Specifically, it studies the relation between the noise skewness and the false-alarm rate, power, and the probabilities of error. The Cornish-Fisher expansion for the sample mean is employed so that performance of the detector can be expressed in terms of the noise skewness. In addition, to simplify the analysis, the noise skewness is assumed to be small so that reasonable approximations can be obtained. Furthermore, we look for a detection scheme which is less sensitive to the noise skewness. The scheme studied here is a modified linear detector with some correction terms which eliminates the effects of skewness.

The following section considers the performance sensitivity of the linear detector to noise skewness. Section III proposes a detection scheme which may be less sensitive to noise skewness. Section IV gives some examples of skewed noise densities and plots the associated Neyman-Pearson optimal test statistics. Section V concludes this paper.

II. Sensitivity of the Linear Detector to Noise Skewness

It is well known that the linear detector is the Neyman-Pearson optimal detector when the underlying noise is Gaussian. However, when the noise density is not Gaussian due to skewness, the performance of the linear detector may deteriorate. Little work has been done on relating performance of the linear detector to noise skewness, partly because the conventional skewness measure [4][5] does not appear explicitly in the likelihood ratio or the test statistics. The main purpose of this section is to study the effect of noise skewness on the performance of the linear detector. In order to facilitate the study, the underlying noise will be assumed to be only slightly skewed and nearly Gaussian. This assumption enables us to use the first few terms of the Cornish-Fisher expansion [7] and obtain a good approximation to the test statistic.

The following canonical binary hypothesis testing problem will be considered:

$$H_0: X_i = N_i \qquad i=1,2,...N$$
$$H_1: X_i = N_i + s, \; s>0$$

It is assumed here that, under H_0, the p.d.f. of X_i is $f(x_i)$ with zero mean, known variance σ^2 and known third-order central moment μ_3, where μ_3 is assumed to be small. Furthermore, the observation sequence $\{X_i\}$ is supposed to be independent and identically distributed. Now, consider the following test statistic

$$T(X) = \overline{X} \triangleq \frac{1}{N} \sum_{i=1}^{N} X_i$$

This T(X) can be written in terms of the Cornish-Fisher expansion, namely,

$$\text{under } H_0: \; T(X) = \frac{\sigma}{\sqrt{N}} Z + \frac{\mu_3}{6\sigma^2 N}(Z^2-1) + O(N^{-3/2})$$

and (1)

$$\text{under } H_1: \; T(X) = \frac{\sigma}{\sqrt{N}} Z + s + \frac{\mu_3}{6\sigma^2 N}(Z^2-1) + O(N^{-3/2})$$

where Z is a standard normal random variable and $O(N^{-3/2})$ stands for the terms which converge to zero faster than $N^{-3/2}$ as N increases.

It is clear that, in order to make the problem tractable, the series in (1) has to be truncated. Since it has been assumed that μ_3 is small, the truncated series will be a good approximation to T(X). Now, define two random sequences $\{Y_0(N)\}$ and $\{Y_1(N)\}$ by

$$Y_0(N) = \frac{\sigma}{\sqrt{N}} Z + \frac{\mu_3}{6\sigma^2 N}(Z^2-1)$$

and (2)

$$Y_1(N) = \frac{\sigma}{\sqrt{N}} Z + s + \frac{\mu_3}{6\sigma^2 N}(Z^2-1)$$

Then, it can be shown that $\{Y_0(N)\}$ and $\{Y_1(N)\}$ converge to T(X) almost surely under H_0 and H_1 respectively. Hence the performance of the detector which uses T(X) as the test statistic can be evaluated approximately via the statistics of $\{Y_0(N)\}$ and $\{Y_1(N)\}$, provided that N is sufficiently large.

Let $A=\mu_3/6\sigma^2 N$ and $B=\sigma/\sqrt{N}$, then Eq. (2) becomes $Y_0(N)=AZ^2+BZ-A$ and $Y_1(N)=AZ^2+BZ-A+s$.

Note that Z is a standard normal random variable. Thus the probability density functions of Y_0 and Y_1 are given by

$$f_{Y_0}(y)\begin{cases} = \frac{1}{\sqrt{2\pi}} Q^{-1}(y)\left\{\Phi\left(-\frac{B}{2A}\right)\exp\left[-\frac{1}{2}\left(\frac{Q(y)+B}{2A}\right)^2\right]+\Phi\left(\frac{B}{2A}\right)\exp\left[-\frac{1}{2}\left(\frac{Q(y)-B}{2A}\right)^2\right]\right\} & \text{if } y \geq -A-\frac{B^2}{4A} \\ =0 & \text{otherwise} \end{cases} \tag{3}$$

where

$$Q(y) = \sqrt{B^2+4A^2+4Ay} \tag{4}$$

and

$$f_{Y_1}(y) = f_{Y_0}(y-s) \tag{5}$$

The false-alarm rate α is given by

$$\alpha = \int_{T_0}^{\infty} f_{Y_0}(y)\,dy \tag{6}$$

where T_0 is the threshold. From Eq. (4), it follows that

$$dQ(y) = 2A\,Q^{-1}(y)\,dy \tag{7}$$

Substituting Eqs. (7) and (3) into Eq. (6) yields

$$\alpha = \int_{Q(T_0)}^{\infty} \Phi\left(-\frac{B}{2A}\right)\frac{1}{2A\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{Q+B}{2A}\right)^2\right]dQ$$

$$+\int_{Q(T_0)}^{\infty} \Phi\left(\frac{B}{2A}\right)\frac{1}{2A\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{Q-B}{2A}\right)^2\right]dQ$$

Thus

$$\alpha = \Phi(-\frac{B}{2A})\Phi(-\frac{T_Q+B}{2A}) + \Phi(\frac{B}{2A})\Phi(-\frac{T_Q-B}{2A}) \tag{8}$$

where

$$T_Q=Q(T_0)$$

Similarly, the power β is given by

$$\beta=\Phi(-\frac{B}{2A})\Phi(-\frac{Q(T_0-s)+B}{2A}) + \Phi(\frac{B}{2A})\Phi(-\frac{Q(T_0-s)-B}{2A}) \tag{9}$$

If the skewness is defined as

$$s_k=\mu_3/\sigma^3$$

then, it is obvious that both α and β are dependent upon the noise skewness, s_k, via the variables $B/2A$, $T_Q/2A$ and $Q(T_0-s)/2A$. In particular,

$$\frac{T_Q+B}{2A} = \sqrt{1+\frac{9N}{s_k^2} + \frac{6NT_0}{\sigma s_k}} \pm 3\sqrt{N}/s_k \tag{10}$$

Figure 1 gives a set of curves which depicts the normalized false-alarm rate, power, and probability of error as functions of the noise skewness. It is assumed that the underlying noise density is positively skewed. It can be seen that both the false-alarm rate and the probability of error are monotone increasing functions of the noise skewness s_k. On the other hand, the power stays nearly unchanged. This illustrates the performance deterioration of the linear detector when the underlying noise is skewed.

In Eq. (3), by use of the L'Hospital's rule, it can be shown that

$$\lim_{s_k\to 0} f_{Y_0}(y) = g(y) \tag{11}$$

where

$$g(y) = \frac{1}{\sqrt{2\pi B^2}}\exp[-\frac{1}{2}(\frac{y}{B})^2] \quad \text{for} \quad -\infty<y<\infty$$

and thus

$$\lim_{s_k\to 0}\alpha = \Phi(-\frac{T_0}{B}) \quad \text{and} \quad \lim_{s_k\to 0}\beta = \Phi(-\frac{T_0-s}{B}) \tag{12}$$

which can also be easily derived from Eqs. (8) and (9) by L'Hospital's rule.

Furthermore, it is interesting to note that the false-alarm rate and the power are bounded. By observing that

$$\Phi(x)-\Phi(y) < \Phi(x-y) < \Phi(x+y) < \Phi(x) + \Phi(y), \quad y\geqslant 0$$

and

$$\Phi(x+y) < \Phi(x-y), \quad y<0$$

one can show that

$$\Phi(-\frac{T_Q+B}{2A})\leqslant \alpha \leqslant \Phi(-\frac{T_Q-B}{2A}) \tag{13}$$

and

$$\Phi(-\frac{Q(T_0-s)+B}{2A}) \leqslant \beta \leqslant \Phi(-\frac{Q(T_0-s)-B}{2A}) \tag{14}$$

In fact, if $T_0>s-A=s-\frac{\sigma s_k}{6N}$, the power β is upper bounded by 0.5.

III. A Modified Linear Detector

In Section II, the sensitivity of the linear detector to the noise skewness was discussed. Figure 1 illustrated how the performance of the linear detector deteriorates when the underlying noise distribution is skewed. It was shown that both the false-alarm rate and the probability of error increases as the amount of noise skewness increases. Now a heuristic question may be asked: Are there "good" detectors which are less sensitive to noise skewness and provide "satisfactory" performance? To answer this question generally is quite involved. Thus, in this article, we restrict ourselves to a special case, namely, to the case when the underlying noise distribution is slightly non-Gaussian due to skewness.

An intuitively reasonable detection scheme should possess at least the following properties:

(a) It should have a "reasonably good" (nearly optimal in some sense) performance for the presumed input model.

(b) It should be fairly insensitive to small deviations from the prescribed input model assumptions.

The Neyman-Pearson optimal detector maximizes the power for a given false-alarm rate. However, evaluation of its test statistic requires a complete knowledge of the underlying noise density function. Therefore, it is possible that deviations from the presumed model would result in impairment of the detector performance. In fact, as shown in the previous section, when the underlying noise density is slightly non-Gaussian due to skewness, the performance of the linear detector does deteriorate. A possible solution to this problem is to modify the optimal detector in such a way that the effect of the noise skewness can be reduced. In this section, the modification of the linear detector is derived by using the Cornish-Fisher expansion for the sample mean and then introducing a mean-squared term so that the terms involving the lowest power of skewness measure can be eliminated in the expansion. Thus, all other terms involving skewness are of higher power. Now, if one assumes that the skewness is small, the terms involving higher powers of skewness measure are neglegible. Hence, one can expect that this modification of the linear detector would reduce the effect of noise skewness on the detector performance.

The following hypothesis testing problem is considered:

$$\begin{aligned} H_0&: X_i = N_i \\ & \qquad\qquad\qquad i=1,2,\ldots,N \\ H_1&: X_i = N_i + s, \quad s>0 \end{aligned}$$

As in section II, it is assumed here that the observation sequence $\{X_i\}$ is independent and identically distributed. Under H_0, the p.d.f. of X_i is $f(x_i)$. Without loss of generality, one can also assume that, under H_0, $E(X_i)=0$ and $E(X_i^2)=\sigma^2$.

Let the modified linear detector have the following test statistic

$$T(X) = \overline{X} + a\left(\overline{X}^2 - \frac{\sigma^2}{N}\right) \tag{15}$$

where

$$\overline{X} \triangleq \frac{1}{N}\sum_{i=1}^{N} X_i$$

and the parameter a will be determined so that the effect of noise skewness may be reduced via the introduction of the term $(\overline{X}^2 - \frac{\sigma^2}{N})$. The Cornish-Fisher expansion for $\overline{X}$ is given by

$$\text{under } H_0:\ \overline{X} = \frac{\sigma}{\sqrt{N}} Z + \frac{\mu_3}{6\sigma^2 N} (Z^2-1) + 0(N^{-3/2}) \tag{16}$$

and

$$\text{under } H_1:\ \overline{X} = \frac{\sigma}{\sqrt{N}} Z + s + \frac{\mu_3}{6\sigma^2 N}(Z^2-1) + 0(N^{-3/2}) \tag{17}$$

where all notations are defined similarly to those in Section II. Substituting Eq. (16) into Eq. (15) yields

$$T(X) = \frac{\sigma}{\sqrt{N}} Z + \frac{\mu_3}{6\sigma^2 N} (Z^2-1) + a\,\frac{\sigma^2}{N} (Z^2-1) + 0(N^{-3/2}) \tag{18}$$

Now, the parameter a can be choosen so that the terms involving μ_3 are eliminated. Thus, if $\mu_3 < 1$, the statistics of T(X) can be expected to be less affected by the noise skewness. Hence, from Eq. (18),

$$a = -\frac{\mu_3}{6\sigma^4} \tag{19}$$

and then, by Eq. (15),

$$T(X) = \overline{X} - \frac{\mu_3}{6\sigma^4} (\overline{X}^2 - \frac{\sigma^2}{N}) \tag{20}$$

In practice, the third-order central moment μ_3 may not be known exactly. Therefore, some learning procedure or adaptive scheme may be needed to obtain an estimate. Then the employment of this detection scheme may be carried out with that estimate.

To evaluate the performance of the detection scheme, one has to compute the statistics of the test statistic: Note that by the Central Limit Theorem, $\overline{X}$ converges in law to a normal random variable $\sigma Z/\sqrt{N}$, where $Z \sim N(0,1)$, provided that

$$\sigma^2 > m > 0$$

and

$$\mu_3 < M$$

where m and M are constants. Now that T(X) is a continuous function of $\overline{X}$, and $\overline{X}$ converges in law to $\sigma Z/\sqrt{N}$, it can be shown [8] that T(X) converges in law to a random variable T, where

$$T = \frac{\sigma}{\sqrt{N}} Z - \frac{\mu_3}{6\sigma^2 N} (Z^2-1) \tag{21}$$

Thus, when N is large enough, the statistics of T(X) can be well approximated by those of T. It is easy to see that, from Eq. (21), the p.d.f. of T is given by

$$f_T(t) \begin{cases} = \sqrt{\dfrac{N}{2\pi\sigma^2}}\, R^{-1}(t) \left\{ \Phi(\dfrac{1}{\sigma_s}) \exp[-\dfrac{1}{2}(\dfrac{1-R(t)}{\sigma_s})^2] + \Phi(-\dfrac{1}{\sigma_s}) \exp[-\dfrac{1}{2}(\dfrac{1+R(t)}{\sigma_s})^2] \right\} & \text{if } t \leqslant \dfrac{1.5\sigma}{s_k} + \dfrac{\sigma s_k}{6N} \\ = 0 & \text{otherwise} \end{cases} \tag{22}$$

where

$$R(t)=\sqrt{1+\frac{s_k^2}{9N}-\frac{2s_k t}{3\sigma}}\ ,\quad \sigma_s = s_k/3\sqrt{N}$$

and $s_k = \mu_3/\sigma^3$, as was defined in Section II.

Then, the false-alarm rate α can be evaluated as follows: With a given threshold T_0,

$$\alpha \approx \int_{T_0}^{\infty} f_T(t)\ dt$$

Now, if $T_0 \geqslant 1.5\sigma/s_k + \sigma s_k/6N$, $\alpha \equiv 0$ which is a case of little interest. On the other hand, if $T_0 < 1.5\sigma/s_k + \sigma s_k/6N$, then

$$\alpha \approx \int_{T_0}^{1.5\sigma/s_k+\sigma s_k/6N} f_T(t)\ dt$$

$$= \Phi(\frac{1}{\sigma_s})\left[\Phi(\frac{1}{\sigma_s}) - \Phi(\frac{1-R(T_0)}{\sigma_s})\right] + \Phi(-\frac{1}{\sigma_s})\left[\Phi(-\frac{1}{\sigma_s}) - \Phi(-\frac{1+R(T_0)}{\sigma_s})\right] \quad (23)$$

Similarly, it can be shown that the power β can be approximated by

$$\beta \approx \Phi(\frac{1}{\sigma_s})\left[\Phi(\frac{1}{\sigma_s}) - \Phi(\frac{1-R(T_0-s)}{\sigma_s})\right] + \Phi(-\frac{1}{\sigma_s})\left[\Phi(-\frac{1}{\sigma_s}) - \Phi(-\frac{1+R(T_0-s)}{\sigma_s})\right], \quad (24)$$

as long as $T_0 < 1.5\sigma/s_k + \sigma s_k/6N$.

In addition, it should be noted that, if s_k is small (say 0.5) and N is large (say 25), then

$$\sigma_s = \frac{1}{30} << 1$$

Thus the expressions in Eqs. (23) and (24) can be further simplified:

$$\alpha \approx 1 - \Phi(\frac{1-R(T_0)}{\sigma_s}) \quad (25)$$

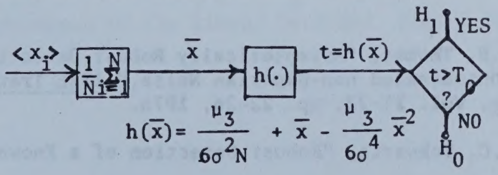
and

$$\beta \approx 1 - \Phi(\frac{1-R(T_0-s)}{\sigma_s}) \quad (26)$$

respectively. Now, combining Eq. (25) and (26) gives

$$\beta = 1 - \Phi(\frac{1-\sqrt{\frac{2s_k}{3\sigma}s + (1-\sigma_s\Phi^{-1}(1-\alpha))^2}}{\sigma_s}) \quad (27)$$

The structure of this modified linear detector is shown in the following diagram:



$$h(\overline{x}) = \frac{\mu_3}{6\sigma^2 N} + \overline{x} - \frac{\mu_3}{6\sigma^4}\overline{x}^2$$

To complete the discussion of this modified linear detector, we have to evaluate its performance and compare it to the linear detector. In this section, we examine the performance of both detectors by plotting the rate of change in false-alarm rate versus the skewness measure and also the ROC curves. It is seen in Figure 2 that, for the linear detector, the false-alarm rate increases while, for the modified linear detector, the false-alarm rate decreases as the skewness measure increases for constant threshold. Moreover, it has been found that the power stays nearly the same for both detectors. In applications such as radar, the received signal may be a pulse-modulated sinusoid and $P(H_0)>>P(H_1)$. In general, if it is desired to keep the total expected cost low, the false-alarm rate will be strictly required to be small. Thus the decrease in the false-alarm rate can really be regarded as a performance improvement. On the other hand, if it is desirable to keep the false-alarm rate fixed, then, in these cases, some adaptive schemes may be needed to adjust the threshold.

Furthermore, Figure 3 gives a set of ROC curves with $s_k=0.5$ and $d\overset{\Delta}{=}s/\sigma$ as a parameter. In contrast, Figure 4 is a similar set of ROC curves for the linear detector. Figure 1-4 can be regarded as indications that both the modified linear detector and the linear detector are reasonaly robust for slightly non-Gaussian noise due to skewness.

IV. Examples:

The density function given in Eq. (3) is used here as an example of a skewed density function. This density function is plotted in Figure 5 together with a normal density function which has zero mean and the same variance. Figure 6 depicts their associated Neyman-Pearson optimal statistics. The results of implementing either the linear detector or the modified linear detector are obtainable from the previous section and hence are not repeated here.

V. Conclusion:

The performance of the linear detector is evaluated with respect to the underlying noise skewness. Some deterioration occurs as the noise becomes skewed. The significance of this performance deterioration will depend on application requirements. However, a modified linear detector is designed in this paper so that the noise skewness will have less effect on the detector performance. In fact, when the underlying noise is nearly Gaussian, it is shown in Section III that the performance is improved and is less sensitive to the skewness. Some generalizations of the analysis used in this paper may be possible; for example, examination and modification of the sign detector and its generalized version with respect to the underlying noise skewness.

References:

[1]. S.A. Kassam and J.B. Thomas, "Asymptotically Robust Detection of a Known Signal in Contaminated non-Gaussian Noise," IEEE Trans. Information Theory, Vol. IT-22, pp. 22-26, 1976.

[2]. R.D. Martin and S.C. Schwartz, "Robust Detection of a Known Signal

in Nearly Gaussian Noise," IEEE Trans. Information Theory, Vol. IT-17, pp. 50-56, 1971.

[3]. S.A. Kassam and J.B. Thomas, "Generalizations of the Sign Detectors Based on Conditional Tests," IEEE Trans. Communications, Vol. COM-24, pp.481-487, 1976

[4]. M.G. Kendall and A. Stuart, The Advanced Theory of Statistics. Vol. I, New York: Hafner, 1963

[5]. W.G. Nichols and J.D. Gibbons, "Parameter Measure of Skewness," Commun. Statist., Vol. B-8, pp 161-167, 1979.

[6]. N.J. Johnson, "Modified t-Test and Confidence Intervals for Asymmetrical Populations," JASA, Vol.73, pp. 536-544, 1978.

[7]. E.A. Cornish and R.A. Fisher, "Moments and Cumulants in the Specification of Distributions," Paper #31 in R.A. Fisher, Contributions to Mathematical Statistics. New York: John Wiley, 1950.

[8]. P. Billingsley, Probability and Measure. New York: John Wiley, 1979.

ACKNOWLEDGEMENT

This research is supported in part by the National Science Foundation under Grant ECS-79-18915, and in part by the Office of Naval Research under Contract N0014-81-k-0146.

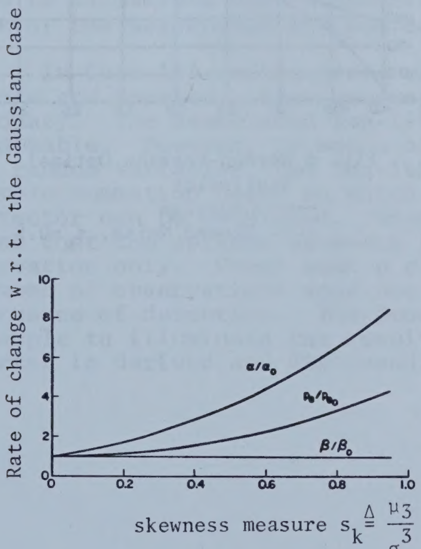


Fig. 1 Performance of the Linear Detector

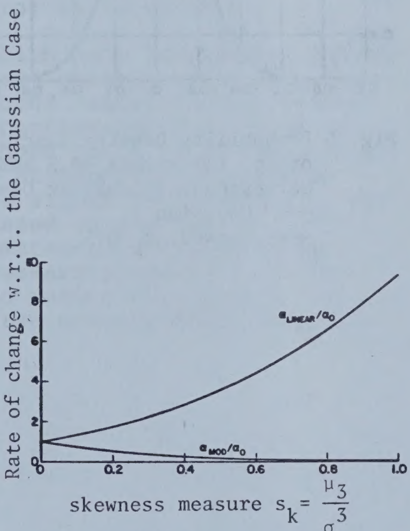
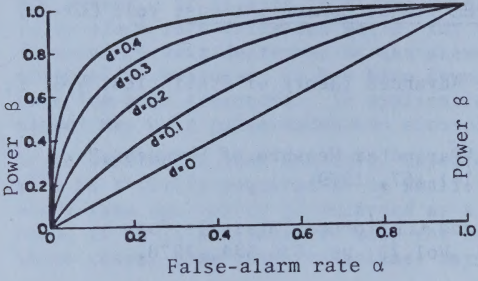
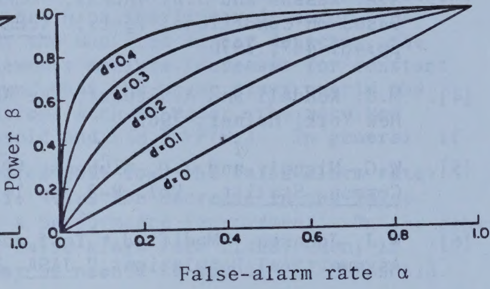


Fig. 2 Comparison of the linear and the modified linear detector



ROC Curves for the Modified Linear Detector, $s_k=0.5$



ROC Curves for the Linear Detector

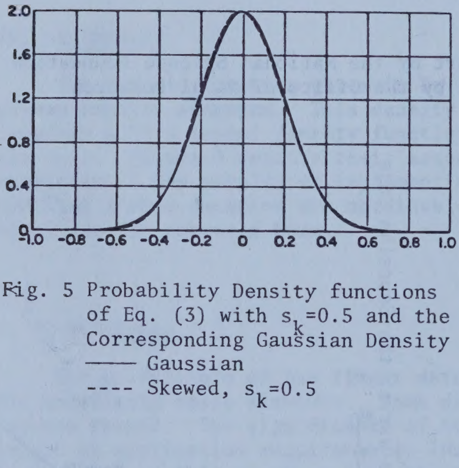


Fig. 5 Probability Density functions of Eq. (3) with $s_k=0.5$ and the Corresponding Gaussian Density
—— Gaussian
---- Skewed, $s_k=0.5$

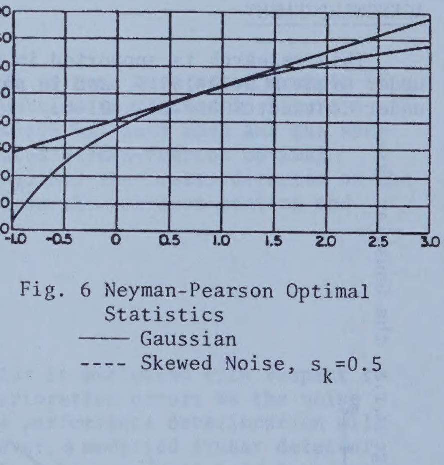


Fig. 6 Neyman-Pearson Optimal Statistics
—— Gaussian
---- Skewed Noise, $s_k=0.5$

CASE STUDY OF LIKELIHOOD RATIO DETECTOR

C.C. LEE
Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, Illinois, 60201

SUMMARY

The detection problem of deterministic signal in additive noise is considered. In each selected case, the log-likelihood ratio is investigated.

In Case I, the noise process is assumed to be stationary and white. Under some regularity conditions, it is shown that, asymptotically, the statistical character of the log-likelihood ratio differs under two hypotheses only in the sign of its mean. When the signal is small, we illustrate how an approximation to the log-likelihood ratio can result in the well-known locally optimum detection scheme. When the signal is unknown, both in magnitude and in sign, a maximum likelihood estimate is needed to establish the likelihood ratio; a chi-squared detector is derived in this manner, as an example.

In Case II, the noise is assumed to be a stationary Markov process. Although the independent nature of input observations is no more available, the Markovian assumption makes it possible to simplify the log-likelihood. The Ornstein-Uhlenbeck process which is zero-mean Gaussian Markov is taken as an example to illustrate the mathematical treament of the likelihood ratio; the associated optimum detector is developed.

In Case III, noise processes which have independent increments are treated. Such processes are neither white nor stationary. The associated log-likelihood ratio is not readily tractable. However, by means of a nonsingular transformation of random variables, the log-likelihood ratio is reduced to a simple summation based on which the Neyman-Pearson optimum detector can be developed. When the signal is a constant, we find that the optimum detector is based on the very first observation only. Under such a circumstance, increasing the number of observations does not necessarily improve the performance of detection. The Brownian motion is taken as an example to illuminate the results; the associated optimum detector is derived and discussed.

STEADY-PACED-OUTPUT AND FRACTIONAL-ON-LINE ALGORITHMS ON A RAM

GLENN K. MANACHER
Department of Information Engineering and
Computer Center
University of Illinois
Chicago, Illinois 60680

Keywords: Analysis of Algorithms, Computational Complexity

A folklore non- "definition" of a "real-time" algorithm is an algorithm which processes an input sequence $a_1 \ldots a_n$ in time $\theta(n*h(n))$, producing outputs $z_1 \ldots z_n$ quickly enough so that the time between the emission of outputs is upper-bounded by $h(n)$, and the time between the initiation of the algorithm and the emission of the first output z_1 is likewise the upper-bounded by $h(n)$. In the usual formulation, z_i is boolean: z_i = true signifies acceptance of $a_1 \ldots a_i$.

The above "definition" is of course not that of an authentic real-time algorithm because it does not require the algorithm to be on-line. Nevertheless, algorithms which satisfy the above constraint are of interest in themselves. We shall show in this paper that such algorithms are quite easy to construct. A more useful definition is that the time between outputs as well as the time between the initiation of the algorithm and the first output should be upper-bounded by $ch(n)$, where $c \geq 1$ is a constant for each algorithm. When this condition is satisfied, we say that the algorithm is steady-paced.

The fact that it is easy to find and construct steady-paced algorithms (SPA's) is due to their intimate connection with a weak variant of on-line algorithms which we shall call fractional on-line algorithms.

In what follows, we shall assume that an algorithm reads inputs $a_1 \ldots a_n$ and writes outputs $z_0 z_1 \ldots z_n$, with z_0 simply a marker that is emitted as soon as the algorithm begins. Furthermore, we do not require that z_j be boolean, and thus place no interpretation on z_j as connoting the "acceptance" of $a_1 \ldots a_j$. We also assume without loss of generality that $h(n)$ is a nondecreasing function of n.

Definition 0. The pace of an algorithm is the maximum time between two outputs z_j and z_{j+1} for $0 \leq j < n$. (Note that a large value indicates a slow pace).

Definition 1. A steady-paced algorithm (SPA) is an algorithm requiring time $nh(n)$ to proceed to completion, whose pace is $ch(n)$ for some constant c defined for that algorithm.

* Supported in part by the NSF under grant # MCS-7819709

Definition 2.1. A weak fractional on-line algorithm is an algorithm with the property that if inputs $a_1 \ldots a_i$ have been read, then outputs $z_1 \ldots a_{\lfloor if \rfloor}$ have been written . f is a constant characteristic of the algorithm and $0 < f \leq 1$. f is referred to as the output/input fraction or simply the fraction.

Definition 2.2. A fractional on-line algorithm (FOLA) is a weak fractional on-line algorithm with the additional property that if the writing of outputs $z_1 \ldots z_{\lfloor nf \rfloor}$ requires (no more than) time t, then the writing of $z_1 \ldots z_j$ for $j > \lfloor nf \rfloor$ requires (no more than) time $tj/\lfloor nf \rfloor$.

From here on, we shall say "requires time t" when we mean "requires no more than time t.

Definition 2.3. An algorithm that is both a FOLA and a SPA will be called a steady-paced fractional on-line algorithm (SFA). Intuitively, it is an algorithm that issues outputs regularly, at the same time maintaining an output/input ratio of ϕ, where $f < \phi \leq 1$ for some fixed f.

Definition 3. The s-average pace for a FOLA with fraction f requiring time $nh(n)$ is defined as $h(n)/f$. ("s" stands for "string).

We emphasize that $h(n)$ is a constant only when the algorithm runs in linear time. In most cases of interest, on-line algorithm are linear-time. By contrast, Preparata [P] has devised a real-time algorithm for finding the convex hull of a set of points on the plane. If $a_1 \ldots a_n$ is a sequence of such points, then Preparata's algorithm writes a boolean sequence $z_1 \ldots z_n$ such that z_j is true iff a_j is on the convex hull of $a_1 \ldots a_j$. Since it is known that the convex hull determination requires $\Omega(n \log n)$ comparisons the pace $\Theta(\log n)$ is optimal for this problem. Some of our constructions will be significant specifically in the nonlinear case. In particular, if $h(n)$ is not a constant, then the SFA we derive from a FOLA (lemma 1b) will have a constant output/input fraction.

Definition 4. For an algorithm, the dynamic fraction ϕ is defined as j/i, where z_j is the most recently written output and a_i is the most recently read input.

In this paper, we shall provide examples both of FOLAs that cannot be on-line, and of on-line algorithms that probably cannot be converted to real-time. For all of them, we shall provide a general lemma that assures that all of them can be converted to a SFA. Moreover, each of these conversions will be "natural" in the sense that if algorithm A is converted to algorithm B, then B will perform the same sequence of operations as A except for additional instructions used to control the pace of the input. The key lemmas are as follows.

Lemma 1a. For every FOLA requiring time $nh(n)$, with fraction f, having s-average pace $p = h(n)/f$, there can be constructed a SPA with pace p and fraction $f/h(n)$ that will run in time $n(h(n) + k)$.

Lemma 1b. For every FOLA requiring time $nh(n)$, with fraction f, having s-average pace $p = h(n)/f$, there can be constructed a SFA with pace $p' = 2(h(n) - 1)/f$ and fraction $f/2$ that will run in time $n(h(n) + k')$, where k and k' are constants less than 1 for each new algorithm.

Note that lemma 1b assures us of a SFA even when the original FOLA is nonlinear. Note also that for a linear-time FOLA, the SPA created by lemma 1a is also an SFA.

The proof of lemma 1a will be facilitated by means of a few more definitions.

Definition 5.1. A set of clock instructions σ in an algorithm A is a subset of the instructions in A satisfying the property that the number of elementary instructions executed between the execution of any pair of instructions in σ is not greater than some constant d defined for the algorithm.

We abbreviate "clock instruction" c.i.

Definition 5.2. By a cycle we mean the maximum time delay d between executions of instructions in σ.

We shall use the cycle as the unit of time in the sequel.

We note as a sidelight that the smaller we can make $|\sigma|$, the faster will be the pace and the larger will be the fraction of the constructions of lemma 1a and 1b.

Definition 5.3. An instruction in σ will be called input-advancing if a read is associated with its execution, and output-advancing if a write is associated with its execution.

Assumption 1. We assume that input-advancing instructions may occur in an arbitrarily long unbroken sequence, i.e., that if m cycles occur, the input may advance by as much as m.

Definition 5.4. Let algorithm B simulate algorithm A, issuing each output at the same time or later than the corresponding output of algorithm A. We will say that algorithm A writes its outputs (into a buffer) and that algorithm B emits its outputs, locating each output left in the buffer by A and writing it onto the output tape.

Proof of lemma 1a. We first note that $h(n)$ is nondecreasing implies that $h(i) \leq h(n)$ for $i \leq n$, so that input i is read by time $ih(n)$ cycles. Therefore output j is written by $jh(n)/f$ cycles. Consequently, if a simulating algorithm waits $h(n)/f$ cycles before emitting an output symbol, each emitted symbol will clearly have already been written. Therefore a FOLA can be simulated by any SPA simply by waiting $h(n)/f$ cycles before emitting each output.

Since the input may advance $h(n)/f$ positions in $h(n)/f$ cycles, the fraction is $f/h(n)$. ☐

The proof of lemma 1b is by means of a construction rather similar to Galil's [G] use of his predictability condition to ensure than an on-line algorithm may be simulated by a real-time algorithm, in particular, his use of a "fast" and "slow" rate. We now introduce the related notion of "deterministic probability" ("d.p."). If an event occurs with d.p. of 1/2 at an instruction, it means that it is executed every other time the instruction is executed. If the d.p. is a rational, e.g., 2/7, the event occurs in a pattern with probability 2/7, e.g., with (yes, no, no, yes, no, no, no) as it cyclic pattern of execution every time the instruction is executed.

Definition 6.1. If q/r is rational and less than 1, then an event occurs with d.p. q/r as defined above.

Definition 6.2. If q/r is rational but is greater than 1, an event may still be said to occur with d.p. q/r . In this case, at the affected instruction, either $\lfloor q/r \rfloor$ or $\lceil q/r \rceil$ occurrences of the event occur, the ratio of the two assuring that on the average the number of events is q/r.

The construction for lemma 1b is then as follows. Without loss of generality, we suppose here that σ contains all the input-advancing and all the output-advancing c.i's. On an input-advancing c.i. we emit an output with d.p. $f/2$. On a non-advancing c.i., we emit an output with d.p. $f/2(h(n) - 1)$. It can be shown easily that the dynamic fraction ϕ will satisfy

$$f/2 \leq \phi \leq 1 \qquad (1)$$

The rest of the proof follows the lines of the proof of lemma 1a. □

Finally, we note that k and k' in lemmas 1a and 1b account for the time necessary to emit any of $z_{\lfloor nf \rfloor} \ldots z_n$ that remain unemitted when the simulated FOLA terminates.

We now give three examples of algorithms for which these ideas apply. All of them happen to be linear-time, so that we will use only the construction of lemma 1a, constructing an SPA that is also an SFA.

1. *The recurrence of longest prefix problems*. Let z_i be the largest integer k less than i for which

$$a_1 \ldots a_k = a_{i-k+1} \ldots a_i \; .$$

This is a slight variant of the Knuth-Morris-Pratt string matching algorithm [KMP]. A minor variant of the version given in [AHU] shows that there is a real-time solution when the input alphabet Σ is finite. For infinite Σ, no real-time solution is known; [G] handles only a related acceptance problem.

Our solution begins with the known on-line solution, shown below.

```
Z(1) := 0;
FOR I :=2 STEP 1 TO N DO
 FOR Q := Z(I - 1) RESET Z(Q) TO 0  DO
  IF A(I) = A(Q + 1) THEN EXIT GOT FI ;
 GOT =>       Z(I) := Q + 1;
 THRU => Z(I) := 0       OD      OD
```

Algorithm 1.

In this algorithm, RESET indicates dynamic resetting of the control variable; EXIT GOT exists the inner loop and proceeds to the exit statement labelled by GOT, and finally the THRU exit statement is entered if the innermost loop terminates because of the header. (The notions of an exit statement originated with Charles Zahn and is explained in Knuth [Kn].)

The SPA created from algorithm 1 results from identifying in algorithm 1 a set σ_1 of clock instructions. Luckily, in this case, we can let σ_1 be just one instruction, the test "A(I) = A(Q + 1)". It is known from [KMP] that no more than 2n of these instructions will be executed. Consequently, h(n) = 2 and p = 2 since f = 1. In order to create the SPA, it is necessary only to emit an output on every other execution of this clock instruction. The algorithm that does this is shown below.

```
Z(1) := 0; COUNT := 0; OUT := 1;
FOR I := 2 STEP 1 TO N DO
  FOR Q := Z(I - 1) RESET Z(Q) TO 0  DO
    COUNT := COUNT + 1;
    IF COUNT = 2 THEN
      COUNT := 0;
      EMIT Z(OUT);
        OUT := OUT + 1  FI;
  IF  A(I) = A(Q + 1) THEN EXIT GOT  FI;
 GOT => Z(I) := Q + 1;
 THRU => Z(I) := 0  OD OD;
FOR F := OUT STEP 1 TO N  DO
 EMIT Z(F)         OD
```

Algorithm 2.

2. <u>The all-maximum embedded palindromes problem</u>. Let $a_1 \ldots a_n$ be a string of characters over a possibly infinite alphabet. Let z_i be a pair of integers (e_i, o_i), representing the center position of palindromes in $a_1 \ldots a_n$. o_i represents the center of an odd-length palindrome centered at position i. e_i represents the center of an even-length palindrome centered at the "interstice" between positions i and i+1. In each case, the value assigned to e_i and to o_i is the length of the palindrome of maximum length centered at that position.

Necessarily, the solution to this problem cannot be on-line, since information beyond a_i is necessary to determine z_i. An algorithm was given by the author [MAN] to determine $y_1 \ldots y_n$ in linear-time on-line where $y_i = 1$ iff a_i is the <u>end</u> of an even palindrome; i.e., if $a_1 \ldots a_i$ is an even palindrome. (A minor adaptation provides an analogue for odd palindromes.) This algorithm has the property that it is a FOLA for the computation of e_i with fraction 1/2. It can be proved that h(n) = 3. Consequently, a SPA exists for $z_i = e_i$ or, separately, for $z_i = o_i$; either SPA algorithm has p = 6 and f = 1/6. We shall see in the next section that we can do much better: We can reduce p to 4 and f to 1/4 and, more importantly, we can obtain an SPA algorithm for $z_i = (e_i, o_i)$ for which p = 8 and f = 1/4, i.e., in which there is no loss in f and no loss of efficiency, in the sense that 8 = 4 + 4.

3. <u>Finding how a concatenation of nontrivial (length greater than 1) palindromes is actually partitioned into palindromes</u>. Let P_1 denote a nontrivial palindrome, i.e., one of length greater than 1. Galil and

Seiferas [GS] have shown that the recognition problem for P_1^* on-line in linear-time has a solution, albeit a very sophisticated and nontrivial one. The problem of actually marking the boundaries of each palindrome of some decomposition of P_1^* into component palindromes can be achieved by means of a rather minor modification of [GS]. In this construction, z_i is 0 if a_i is not the right-hand edge of a string of nontrivial palindromes, else it has value k, where k is the index of the right edge of the previous palindrome. By choosing σ carefully, GS can be shown to have $h(n) = 5$.

Using lemma 1a, we obtain an equivalent SPA with $h(n) = 5$, $p = 5$, and $f = 1/5$.

We now consider improvements to the coefficients in lemma 1a and 1b when $f < 1$. Let $n*g(n,f)$ denote the maximum number of cycles required by the FOLA to permit the input to reach a_n and the output to reach $z_{\lfloor nf \rfloor}$. It is straightforward to show that output z_j will always be available (that is, will have been computed) after $jg(n,f)/f$ cycles. There results the following analogue to lemma 1a.

Lemma 2a. For every FOLA with fraction f requiring (no more than) time $ng(n,f)$ cycles to reach input a_n and output $z_{\lfloor nf \rfloor}$, there can be created a SPA with pace $g(n,f)/f$ and fraction $f/g(n,f)$ that will run in time $n(h(n) + k'')$. Moreover, this pace is optimal, in that no simulation will yield a faster pace.

Proof: Follows the same lines as the proof of lemma 1a.

The analogue to lemma 1b is as follows. For input-advancing c.i.'s, we advance the output with d.p. $f/2$. For non-advancing c.i.'s, we advance the output with d.p. $f/2(g(n,f) - 1)$. There follows

Lemma 2b: From every FOLA, as in lemma 2a, there may be constructed an SFA with dynamic fraction ϕ satisfying

$$f/2 \leq \phi \leq f(1 + \tfrac{1}{2}(h(n) - g(n,f))/(g(n,f) - 1)) \qquad (2)$$

(The r.h.s. of (2) can be shown easily to satisfy $f/2 <$ r.h.s ≤ 1), and pace

$$p = 2(g(n,f) - 1)/f \qquad (3)$$

We omit the proofs here; they are quite straightforward.

Consider finally the possibility of combining two or more SFA's with different p's and f's into a single SFA. The obvious way to do this is to modify the various SFA's so that they act like cooperating coroutines. There are two obvious bases for doing this. Consider two SFA's one "primed," the other "unprimed", whose current input and output indices (input and output most recently read and written) are resp. (i,j) and (i´,j´). We can get one style of cooperation by requiring that i and i´ be advanced almost collaterally over both of the SFA's, so that at any one time i will equal either i' or $i' + 1$. Or we can do the same thing with the j's. The results are quite different. If the j's are coordinated, then i and i' are related to the common value of j by the simple formulas $i \le j/f$ and $i' \le j'/f'$. In the worst case $i = j/f$ and $i' = j'/f'$. This observations leads easily to the formula

---

*Note that $g(n,f) < h(n) < g(n,f)/f$ if $f < 1$.

$$p_c = g(n,f)/f + g'(n,f')/f' \qquad (4)$$

where p_c is the pace resulting from combining the two FPA's into a single FPA that emits (z_j, z'_j) every p_c clock cycles. (The reader may show that the pace will be slower when i and i' are similarly coordinated.) The scheme immediately generalizes to more than two SFA's, resulting in

Lemma 3. C SFA's may be combined into a single SFA which emits the outputs $(z^1_j, z^2_j, \ldots, z^C_j)$ in a collective "burst" emission for each j, with pace p_C between emissions, where

$$p_C = \sum_{m=1}^{C} g^m(n,f^m)/f^m \qquad (5.1)$$

in which by definition the m^{th} algorithm has fraction f^m and requires time $ng^m(n,f^m)$ to reach input a_n and output $z^m_{\lfloor nf^m \rfloor}$.

The fraction for this scheme is easily shown to be

$$f_C = \min_{1 \le m \le C} (f^m) \qquad (5.2)$$

(5.1) and (5.2) then comprise the proof for

Lemma 4. It is possible to combine SFA's into a coroutine-coordinated scheme without loss of efficiency. □

We use these lemmas to finish our discussion of example 2. For this problem, we have $f = 1/2$. It turns out that $g(n,1/2) = 2$. From (5.1) and (5.2) we obtain $p = 8$, $f = 1/4$ for the combined even and odd maximum palindrome-finding routine.

Finally, we mention that the same constructions leading to lemmas 1b and 2b lead also to

Lemma 5. Lemma 2b may be recast so that it yields an SFA with fraction ef, where e is any positive real number less than 1. However, as $e \to 1$, p becomes unboundedly slow.

Proof: For input-advancing c.i.'s, advance the output with d.p. ef. For non-advancing c.i.'s, advance the output with d.p. $(1-e)f/(g(n,f) - 1)$. The same analysis leads to an SFA with fraction ef and pace $(g(n,f) - 1)/(1-e)f$. □

REFERENCES

1. [AHU.] Aho, Hopcroft, and Ullman. The Design and Analysis of Computer Algorithms. Addison-Wesley, 1974, p. 335.
2. [G.] Galil, Zvi. Real-time Algorithms for String-matching and Palindrome Recognition. Proc. of the Eight Annual ACM Symposium on the Theory of Computing, Hershey, Pa., May 3-5, 1976, pp. 161-173.
3. [GS.] Galil, Zvi, and Seiferas, Joel. A Linear-time Recognition Algorithm for "Palstar." JACM 25, 1, January 1978, pp. 102-111.
4. [Kn.] Knuth, D.E. Structured Programming with Go to Statements. ACM Computer Surveys, 6, 4, December, 1974, pp. 261-302.
5. [KMP.] Knuth, D.E., Morris, J.H., Jr., and Pratt, V.R. Fast Pattern Matching in Strings. SIAM J. of Computing 6, 2, June 1977, pp. 323-350.
6. [MAN.] Manacher, G.K. A New Linear-time "On line" Algorithm for Finding the Smallest Initial Palindrome of a String. JACM 22, 3, July, 1975,
7. [P.] Preparata, F.P. An Optimal Real-time Algorithm for Planar Convex Hulls. Comm. of the ACM 22, 7, July 1979, pp. 402-404.

A HARD PROBLEM FOR NTIME(n^d)

LARRY DENENBERG and HARRY R. LEWIS
Harvard University, Cambridge, Massachusetts

ABSTRACT

The satisfiability problem for $E^*\forall^{d+2}$ formulas of the predicate calculus is shown to be hard for nondeterministic time n^d by reduction from computation by linear time bounded iterative arrays of dimension d. An upper bound of nondeterministic time n^{2d+6} is also established.

INTRODUCTION

Adachi, Iwata, and Kasai have shown that the cat-and-mouse problem, which is known to be complete for deterministic exponential time, can be restricted by limiting the number of mice to yield a problem (the cat-and-2k-mouse problem) which requires deterministic time n^k for solution on a single-tape Turing machine [AIK]. By so doing they answer a long-standing challenge of Hopcroft which demands a problem both solvable in polynomial time and hard for (nontrivial) polynomial time classes. In this paper we respond to the same question in the context of nondeterministic time, that is, we take a problem that is complete for nondeterministic exponential time and restrict it to obtain for any d, a problem hard for nondeterministic time n^d.

The problem to be addressed is that of determining satisfiability of certain syntax-restricted formulas of the predicate calculus. Specifically, we consider prenex sentences without function signs or identity in which the prefix is constrained to be of the form $E \ldots E \forall \ldots \forall$. This class is known as the Schonfinkel-Bernays class; its decidability was first proved in [BS]. Lewis shows in [L] that satisfiability of $E^*\forall^*$ formulas (as we shall call members of this class) is complete for nondeterministic exponential time. In this paper we will show that $E^*\forall^{d+2}$-Satisfiability is hard for nondeterministic time n^d, for each $d > 0$. (Our computation model for nondeterministic time is the standard multi-tape Turing machine; [AIK] consider only one-tape Turing machines in their definition of DTIME(n^k), yielding a smaller class of languages for k=1 and perhaps for all k.)

We consider upper bounds for the same problem, but our best result still leaves an essentially quadratic gap between the upper and lower bounds. [AIK] give no upper bound for their problem, but the best obvious algorithm yields a biquadratic gap between upper and lower bounds.

LOWER BOUNDS

The lower bound result is obtained by coding an arbitrary computation by a linear time-bounded nondeterministic d-dimensional iterative array into a sentence of $E^*\forall^{d+1}$ which is satisfiable if and only if the coded computation is accepting. Our model of iterative arrays is that of [C] and [S]: A synchronized, infinite d-dimensional lattice of identical nondeterministic finite state automata, whose transitions depend on the current state and the states of the two adjacent automata in each dimension. We typically think of each FSA as containing finitely many registers, each of which can hold a single symbol from some finite alphabet. Initially, each FSA is in a distinguished state (the _quiescent_ state) except for the first n automata along the positive axis in dimension number one: each of these contains one symbol of the input in a register. The iterative array _accepts_ if the FSA at the origin enters a distinguished final state, and it _accepts in linear time_ if there is a constant c such that for every accepted string w there is an accepting computation with at most $c\cdot|w|$ transitions. NIA(d) denotes the class of languages accepted in linear time by nondeterministic d-dimensional iterative arrays.

Let L be a language in NIA(d), and let w be any string over the alphabet of L. We will show how to construct an $E^*\forall^{d+1}$ formula of length $O(n \log n)$, where n is the length of w, that is satisfiable if and only if w is a member of L. Since $NTIME(n^d) \subseteq NIA(d)$ [S], it follows that any language in $NTIME(n^d)$ can be reduced to a recognition problem for $E^*\forall^{d+1}$. Because the size of the $E^*\forall^{d+1}$ formula has increased by a logarithmic factor, we will actually have shown that $E^*\forall^{d+1}$-Satisfiability is hard for $NTIME((n/\log n)^d)$. It is immediate that $E^*\forall^{d+2}$-Sat is hard for $NTIME(n^d)$.

Assume that w is accepted in time cn, where c is independent of w.

We will use 2cn+3 existential variables (or constants) $x_{-cn-1}, x_{-cn}, \ldots x_{-1}\ x_0, x_1, \ldots x_{cn+1}$. These variables are used to index times and array elements. We also use d+1 universal variables $t, y_1, \ldots y_d$ and write $\overline{y}$ for the array element whose coordinates are $y_1, \ldots y_d$. We use e_i to denote the basis vector which has 1 in its i^{th} coordinate and 0 elsewhere.

The formula to be constructed will use the following predicates, where for each we give the intended interpretation:

$K_q t\overline{y}$ -- At time t, element $\overline{y}$ is in state q.

$S^i_q t\overline{y}$ -- At time t, element $\overline{y}+e_i$ is in state q.

$P^i_q t\overline{y}$ -- At time t, element $\overline{y}-e_i$ is in state q.

$K'_q t\overline{y}$ -- At time t+1, element $\overline{y}$ is in state q.

(S^i and P^i refer to successor and predecessor in the i^{th} dimension, respectively.) Note that "q" represents a state of the finite control of the FSA elements of the array and is not another variable; we actually have many predicates for each one described above, one for each state of the finite control.

The formula to be constructed is a conjunction of four parts, whose interpretations can be given informally as follows:

1) The K, S, and P predicates correctly represent successor and predecessor relationships in space and time.

2) The computation of the iterative array proceeds in time as specified by the transition relation of its (identical) component FSAs.

3) The initial state of the iterative array is correct; that is, all elements are quiescent at time 0 except those containing the symbols of w.

4) At the end of the computation (time cn) the central FSA of the iterative array is in the accepting state.

Part (4) is easily accomplished we simply have $K_f x_{cn} \overline{0}$, where f is the final state and $\overline{0}$ stands for $x_0, \ldots x_0$. We must use cn+3 explicit conjuncts for part (3) to specify the contents of the elements along the first axis at time 0 (n of them containing the symbols of w and the rest quiescent) and then write $(y_2 = y_3 = \ldots = y_d = x_0 \quad \vee \quad K_s x_0 \overline{y})$ to indicate that at time 0 all other automata are in the quiescent state s. (Recall that the y_i are universally quantified, and also note that we use standard tricks to write the last statement above without using the identity sign.)

For (2) we specify the legal transitions of an automaton by clauses of the form:

$$K_q t\overline{y} \quad \wedge \quad \bigwedge_{j=1}^{d} \left(P^j_{p_j} t\overline{y} \wedge S^j_{s_j} t\overline{y}\right) \quad \Rightarrow$$

$$K'_{q_1} t\overline{y} \quad \vee \quad K'_{q_2} t\overline{y} \quad \vee \quad \ldots \quad \vee \quad K'_{q_m} t\overline{y}.$$

The intended interpretation is that when the element $\overline{y}$ is in state q at time t, and its successor and predecessor along dimension j are in states s_j and p_j respectively, then at time t+1 element $\overline{y}$ must be in one of the states $q_1, q_2, \ldots q_m$ -- a nondeterministic choice. To take care of boundary conditions, we also write

$$\bigwedge_{i=1}^{d} \left(K_s t y_1 y_2 \ldots y_{i-1} x_{-cn-1} y_{i+1} \ldots y_d \quad \wedge \quad K_s t y_1 y_2 \ldots y_{i-1} x_{cn+1} y_{i+1} \ldots y_d\right)$$

to insure that the elements in the outer shell surrounding the computation all remain quiescent.

Finally, consider part (1). We write

$$K_q t y_1 y_2 \ldots y_{i-1} x_j y_{i+1} \ldots y_d \quad \Leftrightarrow \quad S^i_q t y_1 y_2 \ldots y_{i-1} x_{j-1} y_{i+1} \ldots y_d$$

for each j between -cn and cn+1, for each i between 1 and d, and for each state q of the FSAs. We repeat these clauses for the P^i, except with x_{j+1} on the right hand side (and for j between -cn-1 and cn). Here we have enforced the successor and predecessor relationships between K, P and S in each spatial dimension. We also write

$$K_q x_j \overline{y} \quad <=> \quad K'_q x_{j-1} \overline{y}$$

(for each q and each j > -cn-1) to enforce the relationship between K and K'.

It should be clear that any accepting computation of M on w can be used to specify a model satisfying the formula described above. Conversely, any model satisfying this equation can be inspected to determine the states attained by each FSA in the array at any point of a legal computation which ends by accepting. The length of the formula is O(n log n) since each variable x_i requires space O(log n) to write down, all other symbols are of constant size, and each part of the formula contains O(n) atomic formulas (and hence O(n) symbols). The constant of proportionality depends only on d and characteristics of the array: the size of its alphabets, state set, and transition relation.

UPPER BOUNDS

To obtain an upper bound for $E^*\forall^d$-Sat, we consider a procedure which uses the finite controllability of the Schonfinkel-Bernays class: that is if a sentence in this class has a model at all then it has a finite model. The algorithm is perfectly straightforward: guess a model and verify that the given formula is true in that model. A nondeterministic Turing machine M may be constructed to determine the satisfiability of a formula F of $E^*\forall^d$ as follows:

1) Guess a Model. In the first step, M writes on a work tape (the model tape) a list of the atomic formulas that will be considered true.

2) Expand the Formula. The Herbrand expansion of a formula is the conjunction of copies of the formula with the existential variables substituted for the universal variables in every possible combination. M can form the Herbrand expansion of F on another tape by using a counter tape for each universal variable and incrementing them through all possible combinations of existential variables,

making an instantiated copy of F for each combination.

3) Verify the Expansion. M now verifies that the expanded formula is true in the guessed model. It simply scans the expanded formula checking that each conjunct is true, looking up atomic formula on the model tape.

To analyze the time complexity of this algorithm, we recall that in an $E^*\forall^d$ formula of length n there can be only $O(n/\log n)$ existential variables, since we must express an occurrence of such a variable in F by explicitly writing (in binary, say) an index for it. Therefore the number of instantiated copies of F in the Herbrand expansion is $O((n/\log n)^d)$ and the size of the expanded formula is $O(n\cdot(n/\log n)^d)$. The last expression is also a bound on the size of the model tape since a model certainly exists in which each true atomic formula appears somewhere in the Herbrand expansion. Thus, step 1 above requires time $O(n\cdot(n/\log n)^d)$.

The only costs in step 2 beyond writing out the expansion are copying the contents of a counter tape to the expansion tape at each occurrence of a universal variable and updating the counter tapes after writing each copy of the formula. Since the size of a counter tape is clearly bounded by n and there are at most n occurrences of universal variables in the formula, the time for step 2 is $O(n^2\cdot(n/\log n)^d)$.

The largest cost is in step 3, since in the Turing machine model we have no way to search the model tape efficiently. Each instantiated copy of the input formula may contain $O(n)$ atomic formulas, and for each we may have to search the entire model tape. The cost of step 3 is thus $O(((n/\log n)^d)\cdot n\cdot(n\cdot(n/\log n)^d))$, or $O(n^{2d+2}/(\log n)^{2d})$.

By use of a random access machine model such as the RAM or a "Turing machine with random access" [LP] it is possible to implement the algorithm in time $O(n^{d+c})$ where c is a small constant. However, we know of no construction that preserves the lower bound established above in such a model.

SUMMARY

Combining the results of the sections above, we state here the bounds for $E^*\forall^d$-Satisfiability

Upper bound: $\mathrm{NTIME}(n^{2d+2}/(\log n)^{2d})$.

Lower bound: $\mathrm{NTIME}((n/\log n)^{d-1})$.

REFERENCES

[AIK] A. Adachi, S. Iwata, and T. Kasai, Low level complexity for combinatorial games, Proceedings of the Thirteenth Annual ACM Symposium on the Theory of Computing (1981), 228-237.

[BS] P. Bernays and M. Schoenfinkel, Zum Entscheidungsproblem der mathematischen Logik, Math. Ann. 99 (1928), 342-372.

[C] S. N. Cole, Real-time computation by n-dimensional iterative arrays of finite-state machines, IEEE Transactions on Electronic Computers, EC-18 (1969), 349-365.

[L] H. R. Lewis, Complexity results for classes of quantificational formulas, JCSS 21 no. 3 (1980), 317-353.

[LP] H. R. Lewis and C. Papadimitriou, Elements of the Theory of Computation, Prentice-Hall (1981) p. 362.

[S] J I. Seiferas, Linear-time computation by nondeterministic multidimensional iterative arrays, SIAM Journal on Computing 6 no. 3 (1977), 487-504.

TWO IMPLEMENTATIONS OF RECURSIVE PATH ORDERING ON MONADIC TERMS

Pierre LESCANNE

Centre de Recherche en Informatique de Nancy, CO 140, 54037 Nancy, France
and
Laboratory for Computer Sciences, M.I.T, Cambridge, Massachusetts, USA

Abstract: The Recursive Path Ordering was introduced by Dershowitz and Plaisted to prove termination on rewriting systems. In this paper, a new definition is proposed and its implementation is compared with that deduced from Dershowitz and Plaisted definition.

1. Introduction

The recursive path ordering was introduced by Dershowitz [1] and Plaisted [6] as a tool to prove termination of simplification systems or sets of rewriting rules [3]. It is a well-founded monotonic ordering, with the subterm property (each term is greater than its subterms). From Dershowitz definition, it is easy to deduce an algorithm computing the ordering between two terms, as a function, with parameters these two terms, returning value "greater", "less" and "equal". In [4] we proposed another, equivalent, definition: the decomposition ordering yielding a efficient implementation. In this paper, the comparative study of recursive path ordering algorithm (RPO) and decomposition ordering (DO) is restricted to monadic terms.

The second section of this paper presents the two ordering algorithms. The third section studies their behavior when one of the terms is of length 1. The fourth section is a presentation of simulation results in the general case.

2. Presentation of the algorithms

Let F be a set of monadic symbols and $\square$ be a constant, F$\cup\{\square\}$ will be supposed totally ordered with $\square$ less than all f in F. $\square F^*$ is the set of monadic terms on F in a postfixed notation. In the sequel, s, t, u and v denote terms and the other letters denote symbols in F.

2.1 Recursive path ordering on monadic terms

The recursive path ordering is defined by

$$s \overset{*}{<} t$$

iff

$[s=\Box \text{ and } t\neq\Box]$
$\text{or } [s=uf \text{ and } t=vg \text{ and}$
$\{(f=g \text{ and } u\overset{*}{<}v)$
$\text{or } (f<g \text{ and } u\overset{*}{<}t)$
$\text{or } (s\overset{*}{<}v)$
$\text{or } (s=v)\}]$

For example, let F be {a,b,c} with $a<b<c$, we have:

$$\Box bbabb\overset{*}{<}\Box acbba\overset{*}{<}\Box acbbb\overset{*}{<}\Box bcbb$$

Let us prove the first inequality: $\Box bbabb\overset{*}{<}\Box acbba$ if $\Box bbabb\overset{*}{\leq}\Box acbb$ if $\Box bbab\overset{*}{\leq}\Box acb$ if $\Box bba\overset{*}{\leq}\Box ac$ if $\Box bb\overset{*}{\leq}\Box ac$ if $\Box\overset{*}{\leq}\Box ac$.
The recursive algorithm RPO given by Figure 1, in an algol-like language, can be associated to the recursive path ordering.

To compute $R=RPO(\Box bbabb,\Box acbba)$ we have to compute $R_1 = RPO(\Box bbabb,\Box acbb) = RPO(\Box bbab,\Box acb) = RPO(\Box bba,\Box ac)$, which leads to compute $R_2 = RPO(\Box bb,\Box ac)$ and $R_3 = RPO(\Box b,\Box ac)$ and $R_4 = RPO(\Box,\Box ac)=$'<' and because $R_4=$'<', then $R_3=$'<' then $R_2=$'<' then $R_1=$'<', we conclude than R is '<'. Figure 2 gives a non recursive version of RPO. In the recursive version, indeed, after the three internal calls to RPO, named "call 1", "call 2", and "call 3", the value returned is the value yielded by these internal calls, except for "call 2" and "call 3" when they return '='. In that case, all the next calls are "call 1". Therefore, in order to know the value the program must yield, only the name of the

Fig. 1. Algorithm based on the recursive path ordering

```
RPO=proc(s,t:monadic_term) returns(char)
% RPO returns only '<', '=' or '>' %
        let a=rear(s)
        let b=rear(t)

        case a=b & a=□ then return('=')
        case a=b & a≠□ then return(RPO(rest(s),rest(t))) % call 1 %
        case a<b & a=□ then return('<')
        case a<b & a≠□ then  if RPO(rest(s),t)='<' % call 2 %
                             then return('<')
                             else return('>')
        case a>b & b=□ then return('>')
        case a>b & b≠□ then if RPO(s,rest(t))='>' % call 3 %
                             then return('>')
                             else return('<')
        end RPO
```

innermost call different from 1 must be kept; that is done by the variable "token" in the iterative version. This iterative version can be also thought as a card game similar to "war". The looser is the first who finishes his card stack. An actual token is used to decide between partners when they finish their stack at the same time. In [5] a slightly more complicated version using array and three loops is defined.

Let us note rpo(m,n,v) the average number of comparisons, between symbols, necessary to compare two terms with lengths m and n (i.e. having m and n symbols of monadic function) on a vocabulary with size v, when the symbols are equiprobable, i.e. each one with a probability 1/v. The asymptotic behavior of rpo is not easy to describe, here we give in section 3 the analysis in the case m=1 and in section 4, some simulation results.

2.2 Decomposition ordering on monadic terms

To define the decomposition ordering we let us first explain the *simple decomposition* of a term s other than □; it is a triple [f_s;s';s"], where f_s is the maximum symbol in s, and where s' and s" are terms in □F* such that s" = □σ and s = s'$f_s\sigma$, where σ is a postfix of s, not containing f_s.

Fig. 2. Non recursive version of the algorithm based on the recursive path ordering

```
IT_RPO = proc(s,t:monadic_term) returns(char)
% IT_RPO returns only '<', '=' or '>' %
        token:char := '='
        while true do
                case s = □ & t = □ then return(token)
                case s = □ & t ≠ □ then return('<')
                case s ≠ □ & t = □ then return('>')
                case s ≠ □ & t ≠ □ then
                        case rear(s)<rear(t)    then s: = rest(s)
                                                     token: = '>'
                        case rear(s)>rear(t) then    t: = rest(t)
                                                     token: = '<'
                        case rear(s) = rear(t) then  s: = rest(s)
                                                     t: = rest(t)
        end
end IT_RPO
```

The decomposition ordering is then defined as:

$$s \overset{\delta}{<} t$$

iff

$$\begin{aligned} & (s=\square \text{ and } t\neq\square) \\ \text{or } & (f_s<f_t) \\ \text{or } & (f_s=f_t \text{ and } s' \overset{\delta}{<} t') \\ \text{or } & (f_s=f_t \text{ and } s'=t' \text{ and } s'' \overset{\delta}{<} t'') \end{aligned}$$

Here are a few examples of simple decompositions with $F=\{a, b, c\}$ and $a<b<c$: the simple decomposition of □bbabb is [b;□bbab;□], that of □acbba is [c;□a;□bba], that of □acbbb is [c;□a;□bbb], and that of □bcbb is [c;□b;□bb]. If these terms are compared using the decomposition ordering, the result is the same as with recursive path ordering. In fact, the orderings are equivalent. In the above definition, $\overset{\delta}{<}$ is recursively used twice: once to compare s' to t', and once to compare s" to t". For instance, to compare □acbba and □acbbb, □bba and □bbb are compared using their decomposition. Thus we shall define the *(generalized) decomposition* of a term s, other than □, as triple $dec(s)=[f_s;dec(s');dec(s'')]$, where dec(s') and dec(s") are the generalized decomposition of s' and s", the second and the third components of the simple decomposition of s. The generalized decomposition of □ is not a triple, but □ itself. Generalized decomposition can be represented by a labeled binary tree, the *decomposition tree* (Figure 3).

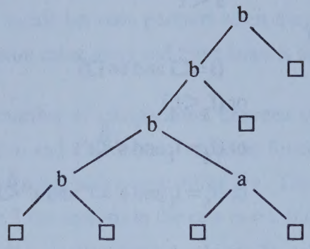
Giving the decomposition tree of a term is similar to building a binary tournament [2] by n successive insertions. The differences are that, in a tournament, an already present symbol is not inserted, while in the decomposition tree, it is, and in the decomposition tree each node is greater than its right son and greater or equal to its left son. J.L. Remy suggested to build a tournament by a bottom-up method on the external right branch, like Françon, Viennot and Vuillemin pagoda [2]. Such an algorithm needs less than $2n-H_n$ comparisons. Figure 4 shows an algorithm for comparing two terms, given their decomposition tree.

Let do(m,n,v) be the average number of comparisons between symbols necessary for comparing two terms of length m and n on a vocabulary of size v, when the symbols are equiprobable; do is studied along with rpo in section 3 and 4.

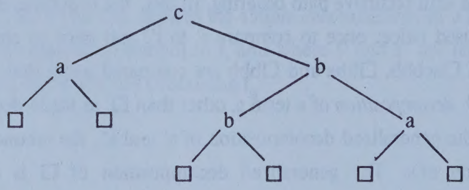
3. Number of comparisons when a term is of length 1

In this section, the average number of comparisons performed by the two algorithms is given, when all the symbols are equiprobable and when one of the two terms, for example the first one is of length 1, i.e. when this term has a unique monadic function symbol; in other words, the values of rpo(1,n,v) and do(1,n,v) are given and some asymptotic approximations are proposed.

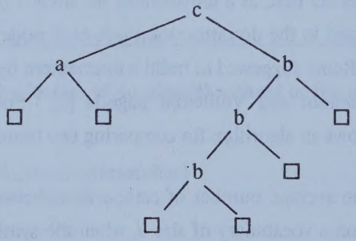
Fig. 3. Some Decomposition Trees



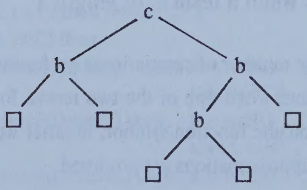
decomposition tree of □bbabb



decomposition tree of □acbba



decomposition tree of □acbbb



decomposition tree of □bcbb

Fig. 4. Decomposition ordering algorithm

```
DO=proc(d1,d2: binary_tree) returns(char)
% DO returns only '<', '=' and '>' %
    st:stack[pair_of[binary_tree]]:=new_stack
    % st is a stack to store the roots of not traversed subtrees %
    while true do
        case d1.label=d2.label='□' then if empty(st) then return('=')
                                                      else d1,d2:=top(st)
                                                           st:=pop(st)
        case d1.label=d2.label≠'□' then d1,d2:=d1.left,d2.left
                                         st:=push(st, (d1.right,d2.right))
        case d1.label>d2.label then return('>')
        case d1.label<d2.label then return('<')
        end % end while %
end DO
```

DO performs a depth first traversing of the two trees, together, until the first difference in the label of the nodes is found.

3.1 Study of rpo(1,n,v)

Let us suppose $s=\square f$ and $t=\square g_n...g_1$. The probability that f is the ith symbol in the ordering of the vocabulary is 1/v. The IT_RPO algorithm proceeds by moving two cursors σ and τ; σ takes either value 1 or 2, (initially 1) and τ takes one of n+1 values in {1,...,n+1} (initially 1). Each recursive call to IT_RPO corresponds to moving one or both cursors. The algorithm terminates, performing one comparison, when $\sigma=2$ or $\tau=n+1$. If the first time $\sigma:=2$ happens when $\tau=k$ ($k\leq n$), then $g_{k-1}...g_1$ are less[1] than f (with probability $(\frac{i-1}{v})^{k-1}$) and g_k is greater[1] than or equal to f (with probability $\frac{v-(i-1)}{v}$). Thus

$$\Pr\{\sigma:=2 \text{ when } \tau=k\}=\sum_{i=1}^{v}\frac{1}{v}\left(\frac{i-1}{v}\right)^{k-1}\frac{v-(i-1)}{v}$$

This is also the probability than IT_RPO will perform k+1 comparisons (k-1 to move τ in k, one to move σ in 2 and 1 to decide).

If σ takes never the value 2, then $g_n...g_1$ are less[1] than f (with probability $(\frac{i-1}{v})^n$). Thus

$$\Pr\{\sigma \text{ does not become } 2\}=\sum_{i=1}^{v}\frac{1}{v}\left(\frac{i-1}{v}\right)^{n}$$

This is also the probability than IT_RPO will perform n+1 comparisons (n to move τ and 1 to decide).

Therefore:

$$rpo(1,n,v)=\left[\sum_{k=1}^{n}(k+1)\sum_{i=1}^{v}\frac{1}{v}\left(\frac{i-1}{v}\right)^{k-1}\left(\frac{v-(i-1)}{v}\right)\right]+(n+1)\sum_{i=1}^{v}\frac{1}{v}\left(\frac{i-1}{v}\right)^{n}$$

Decomposing v-(i-1) and permuting the Σ:

---

[1] For the vocabulary ordering.

$$=\frac{1}{v}\sum_{i=1}^{v}\Big[\sum_{k=1}^{n}(k+1)\Big(\frac{i-1}{v}\Big)^{k-1}-\sum_{k=1}^{n}(k+1)\Big(\frac{i-1}{v}\Big)^{k}\Big]+(n+1)\sum_{i=1}^{v}\frac{1}{v}\Big(\frac{i-1}{v}\Big)^{n}$$

Changing variable k in k−1 in the second sum and grouping:

$$=\frac{1}{v}\sum_{i=1}^{v}\Big[2+\sum_{k=2}^{n}((k+1)-k)\Big(\frac{i-1}{v}\Big)^{k-1}-(n+1)\Big(\frac{i-1}{v}\Big)^{n}\Big]+(n+1)\sum_{i=1}^{v}\frac{1}{v}\Big(\frac{i-1}{v}\Big)^{n}$$

$$=2+\frac{1}{v}\sum_{i=1}^{v}\sum_{k=2}^{n}\Big(\frac{i-1}{v}\Big)^{k-1}$$

$$=2+\frac{1}{v}\sum_{i=1}^{v}\Big(\frac{i-1}{v}\Big)\frac{1-((i-1)/v)^{n-1}}{1-((i-1)/v)}$$

Using integrals to get asymptotic approximations:

$$2+\frac{1}{v}\int_{1}^{v}\Big(\frac{x-1}{v}\Big)\frac{1-((x-1)/v)^{n-1}}{1-((x-1)/v)}\,dx\leq \mathrm{rpo}(1,n,v)\leq 2+\frac{1}{v}\int_{1}^{v+1}\Big(\frac{x-1}{v}\Big)\frac{1-((x-1)/v)^{n-1}}{1-((x-1)/v)}\,dx$$

Changing (x−1)/v into y:

$$2+\int_{0}^{1-1/v} y\,\frac{1-y^{n-1}}{1-y}\,dy\leq \mathrm{rpo}(1,n,v)\leq 2+\int_{0}^{1} y\,\frac{1-y^{n-1}}{1-y}\,dy$$

Decomposing:

$$2+\sum_{k=1}^{n-1}\int_{0}^{1-1/v} y^{k}\,dy\leq \mathrm{rpo}(1,n,v)\leq 2+\sum_{k=1}^{n-1}\int_{0}^{1} y^{k}\,dy$$

Computing the integrals:

$$2+\sum_{k=1}^{n-1}\frac{1}{k+1}\Big(1-\frac{1}{v}\Big)^{k+1}\leq \mathrm{rpo}(1,n,v)\leq 2+\sum_{k=1}^{n-1}\frac{1}{k+1}=1+H_n$$

If v grows faster than n :

$$2+\sum_{k=1}^{n-1}\frac{1}{k+1}-\sum_{k=1}^{n-1}\frac{1}{v}+O((n/v)^2)\leq \mathrm{rpo}(1,n,v)\leq 1+H_n$$

Thus:

$$1+H_n-\frac{n-1}{v}+O((n/v)^2)\leq \mathrm{rpo}(1,n,v)\leq 1+H_n \qquad \text{as } n/v\to 0.$$

3.2 Study of do(1,n,v)

do(1,n,v) is the average number of test necessary to compute DO(s,t) when length(s) = 1, length(t) = n and size of the vocabulary is v. The computation of the probability can be divided into three parts depending on the number of comparisons. It is easy to see that at most three comparisons are necessary because the decomposition tree of s (Figure 5) has exactly three nodes. In the sequel f will denote the ith symbol of the vocabulary (its probability is 1/v).

–If three comparisons are needed (t_3 in Figure 5) then $s=\square f$ and $t=\square f\, g_{n-1}\ldots g_1$, with $g_{n-1},\ldots,g_1$ less than f, with probability $\frac{(i-1)^{n-1}}{v^n}$. Thus

$$P_3=\Pr\{\text{three comparisons are needed}\}=\sum_{i=1}^{v}\frac{(i-1)^{n-1}}{v^{n+1}}$$

–If two comparisons are needed (t_2 in Figure 5) then $s=\Box f$ and $t=\Box g_n \dots g_{k+1} f g_{k-1} \dots g_1$ with $g_{k-1},\dots,g_1$ less than f with probability $(\frac{i-1}{v})^{k-1}$ and $g_n,\dots,g_{k+1}$ less than or equal to f, with probability $(\frac{i}{v})^{n-k}$. Thus

$$P_2 = \Pr\{\text{two comparisons are needed}\} = \sum_{i=1}^{v} \sum_{k=1}^{n-1} \frac{(i-1)^{k-1}\, i^{n-k}}{v^{n+1}}$$

–If one comparison is needed (t_1 in Figure 5)

$$P_1 = \Pr\{\text{one comparison is needed}\} = 1 - P_2 - P_3$$

Therefore:

$$do(1,n,v) = P_1 + 2P_2 + 3P_3 = 1 + P_2 + 2P_3$$

$$= 1 + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} \sum_{k=1}^{n-1} (i-1)^{k-1}\, i^{n-k} + 2\, (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} (i-1)^{n-1}$$

$$= 1 + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} \sum_{k=1}^{n} (i-1)^{k-1}\, i^{n-k} + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} (i-1)^{n-1}$$

Because $i \geq i-1$:

$$do(1,n,v) \geq 1 + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} \sum_{k=1}^{n} (i-1)^{n-1} + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} (i-1)^{n-1}$$

$$do(1,n,v) \geq 1 + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} (n+1)\, (i-1)^{n-1}$$

Using integrals to get asymptotic approximations:

$$do(1,n,v) \geq 1 + (\tfrac{1}{v})^{n+1} \int_1^v (n+1)\, (x-1)^{n-1}\, dx$$

Computing these integrals:

$$do(1,n,v) \geq 1 + (\tfrac{1}{v})^{n+1} (\tfrac{n+1}{n})\, (v-1)^n$$

$$do(1,n,v) \geq 1 + \frac{1}{v} + \frac{1}{nv} + O(n/v^2)$$

On the other hand:

$$do(1,n,v) \leq 1 + (\tfrac{1}{v})^{n+1} \sum_{i=1}^{v} ((n-1)\, i^{n-1} + 2\, (i-1)^{n-1})$$

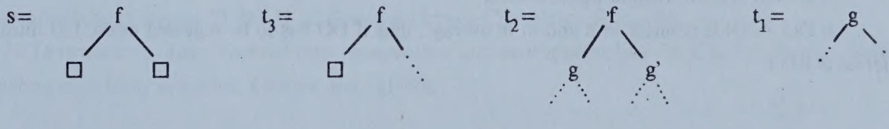
By the same way:

$$do(1,n,v) \leq 1 + \frac{1}{v} + \frac{1}{nv} + O(n/v^2)$$

Thus:

$$do(1,n,v) = 1 + \frac{1}{v} + \frac{1}{nv} + O(n/v^2) \qquad \text{as } n/v \to 0.$$

Fig. 5.



4. Number of comparison in the general case

The following tables give some simulation results for rpo(m,n,v) (on the left-hand side) and do(m,n,v) (on the right-hand side) in each box. Words with size v (here 4 or 16 or 128) were generated, symbols were chosen at random with equal probabilities and statistics on these words were computed. The given numbers are means on samples.

m\n	4		8		16		32		128	
4	5.6	1.8	7.1	1.9	7.6	1.9	8.5	2.0	8.7	2.1
8			10.4	2.3	13.6	2.9	14.3	3.0	14.3	3.0
16					21.5	4.1	25.6	5.0	27.4	4.9
32							43.5	7.7	52.8	9.0
128									190	30.6

Results with a vocabulary of size 4

m\n	4		8		16		32		128	
4	6.0	1.2	7.8	1.2	10.0	1.2	10.6	1.2	11.6	1.2
8			11.6	1.4	15.5	1.4	19.5	1.4	20.2	1.5
16					22.6	1.6	30.3	1.9	35.2	2.0
32							46.3	2.2	64.9	2.8
128									199	7.7

Results with a vocabulary of size 16

These results and others allow the following conjectures:

1) rpo(m,n,v) is always greater than do(m,n,v).

2) rpo(n,n,v) is k.n + O(n/v).

3) do(m,n,v) → 1, when n/v goes to zero.

4) DO + DEC is linear in n and m in average, thus if DO has to be repeated often, DO must be preferred to RPO.

m\n	4		8		16		32		128	
4	6.2	1.0	8.1	1.0	10.8	1.0	13.2	1.0	18.0	1.0
8			11.8	1.0	15.5	1.0	22.4	1.0	30.1	1.1
16					22.6	1.0	31.8	1.1	46.0	1.1
32							46.3	1.2	75.2	1.2
128									191	1.7

Results with a vocabulary of size 128

5. Conclusion

The mathematical and experimental studies give some facts to compare the algorithms based on the decomposition ordering and the recursive path ordering. This suggests that these properties would be still true in the polyadic case keeping as in this paper a statistics based on a equiprobable distribution of the symbols in the terms. In [7], Dershowitz and Zaks propose another approach based on Catalan statistics which takes into account mostly the shapes of the terms.

Acknowledgments I thank Jean-Luc Remy and Marc Shapiro. The first one suggested good improvements and the second helped me during the writing.

6. Bibliography

[1] N. Dershowitz: *Orderings for Term-rewriting Systems,* Proc. 20th Symposium on Foundations of Computer Science, (1979), 123-131.

[2] J. Françon, G. Viennot, J. Vuillemin: *Description and Analysis of an Efficient Priority Queues Representation,* Proc. 19th Symposium of Foundations of Computer Science, (1978), 1-7.

[3] G. Huet, D.C. Oppen: *Equations and Rewrite Rules: a Survey,* in *Formal Languages perspectives and Open Problems,* Ed. Book R., Academic Press, (1980).

[4] Lescanne P.: *Decomposition Ordering as a Tool to prove the Termination of Rewriting Systems,* 7th IJCAI, Vancouver, Canada, (1981).

[5] D. Plaisted: *Well-Founded Orderings for Proving Termination of Systems of Rewrite Rules* Dept of Computer Science Report 78-932, U. of Illinois at Urbana-Champaign, July 1978.

[6] D. Plaisted: *A Recursively Defined Ordering for Proving Termination of Term Rewriting Systems,* Dept of Computer Science Report 78-943, U. of Illinois at Urbana-Champaign, Sept. 1978.

[7] N. Dershowitz, S. Zaks: *Ordered trees enumerations and some applications,* 6th Colloque sur les Arbres en Algebres et en Programmation, Genova, Italy, (1980).

POLYNOMIAL AVERAGE-TIME SATISFIABILITY PROBLEMS

PAUL WALTON PURDOM, JR. and CYNTHIA A. BROWN
Indiana University, Bloomington, Indiana

The time required to solve NP complete problems is a major open question. While no algorithm has been found that solves NP complete problems in subexponential time, some algorithms can solve certain classes of random conjunctive normal form (CNF) problems in polynomial average time.

A class of random CNF predicates is characterized by v, the number of variables from which literals may be formed (giving 2 v literals); and p(v), the probability that a given literal is in a random clause; and t(v), the number of random independently selected clauses in a random predicate. Let $L = \lim_{v \to \infty} vp(v)$. Backtracking takes polynomial time on classes for which (1) $L=0$ and $t(v) \geq \frac{(\ln 2 + \varepsilon)v}{-\ln((v+1)p(v))}$; (2) $L \leq \ln 2$ and $t(v) \geq \frac{(\ln 2 + \varepsilon)v}{-\ln(1-e^{-L})}$; (3) $L \geq \ln 2$ and $t(v) \geq \frac{(\ln 2 + \varepsilon)dv}{L}$ where $\ln(1+d) + d \ln(1+1/d) = 2L$; (4) $L = \infty$ and $t(v) \geq \exp[(2+\varepsilon)vp(v)]$. (In each case, ε is a positive number. For each formula, if ε is taken to be negative and the direction of the inequality on t(v) is reversed, we have a condition for the average time to be exponential.)

An analysis by Goldberg (in Proceedings of the Fourth Annual Workshop on Automated Deduction) showed that problems for which p(v) is constant can be solved in polynomial average time by a simplified version of the Davis-Putnam procedure which essentially uses only the pure literal rule. The exponent of the polynomial gets larger as the constant value of p(v) decreases.

We have developed an algorithm for solving problems for which $t(v) \leq \frac{\ln \ln v}{\ln 3}$ in polynomial time. Together with the above results this means that for sets of problems with p(v) or t(v) either very large or very small there is some algorithm that will solve the problem in the set in polynomial average time.

This work was supported in part by the National Science Foundation under grant number MCS 7906110.

ON UNIVERSALITY OF CONCURRENT EXPRESSIONS WITH SYNCHRONIZATION PRIMITIVES

T. Ito and Y. Nishitani
Department of Electrical Communications
School of Engineering
Tohoku University
Sendai, JAPAN

Abstract: Concurrent expressions are a class of extended regular expressions with a shuffle operator($\|$) and its closure($\textcircled{\|}$).The class of concurrent expressions with synchronization primitives,called synchronized concurrent expressions,is introduced as an extended model of Shaw's flow expressions.This paper discusses some formal properties of synchronized concurrent expressions from a formal language theoretic point of view.It is shown that synchronized concurrent expressions with three signal/wait operations are universal in the sense that they can simulate any semaphore controlled concurrent expressions and they can describe the class of recursively enumerable sets. Some results on semaphore controlled *regular* expressions are also included to give a taste of more positive results.

I. INTRODUCTION

A number of formal approaches to investigate properties of concurrent processes are proposed in the literature.A.C. Shaw[1978] proposed a model called "flow expression". Flow expressions are a class of extended regular expressions with a shuffle operation,its closure,an infinite cycle operation and synchronization primitives.Shaw[1978] discusses the correspondence between flow expressions and concurrent processes and give some elementary properties of flow expressions.Also,Shaw[1979] discusses some relations between flow expressions and other models of concurrent processes.Some formal properties of flow expressions have been studied by Shaw,Araki-Tokura and others from a formal language theoretic point of view,as is mentioned in Shaw[1979]

In this paper we introduce a slightly generalized formalism of flow expressions,called "synchronized concurrent expressions",which is more suitable for our theoretical study on concurrent processes.Concurrent expressions are a class of extended regular expressions with a shuffle operator($\|$) and its closure($\textcircled{\|}$).The class of concurrent expressions with synchronization primitives(like semaphore primitives) is called synchronized concurrent expressions in this paper.We introduce a reasonable class of synchronization primitives defined as concurrent expressions,capable of describing various semaphore systems.Using this formalism we show that synchronized concurrent expressions with three signal/wait semaphore operations are universal in the sense that they can simulate any semaphore controlled concurrent expressions and they can describe the class of recursively enumerable sets.Some latest results on semaphore controlled regular expressions are also included to explain more positive theoretical aspects of these studies.

II. SYNCHRONIZED CONCURRENT EXPRESSIONS

In this section we give the definitions of

(1) concurrent expression
(2) synchronized concurrent expression
(3) semaphore systems and synchronization primitives in concurrent expressions

and some previous results.

2.1 Definition of Concurrent Expressions

Let Δ be a finite set of symbols. Then the set of concurrent expressions over Δ can be defined recursively as follows:

(1) $\phi, \lambda, a(\varepsilon\ \Delta)$ are concurrent expressions

(2) if E and F are concurrent expressions, then $(E), E+F, EF, E^*, E\|F$ and $E^{\textcircled{\scriptsize$\|$}}$ are concurrent expressions.

The semantics of concurrent expressions can be defined as follows:

(1) $|\phi| = \phi$ (ϕ: the empty set)

(2) $|\lambda| = \{\lambda\}$ (λ: the empty word)

(3) $|a| = \{a\}$ ($a\varepsilon\Delta$)

(4) $|(E)| = |E|$

(5) $|E+F| = |E| \cup |F|$

(6) $|EF| = \{xy \mid x\varepsilon|E|, y\varepsilon|F|\}$

(7) $|E^*| = \bigcup_{i=0}^{\infty} |E^i|$, where $E^0 = \lambda$, $E^{i+1} = E^i E$ for $i \geq 0$.

(8) $|E\|F| = \{x_1y_1x_2y_2\ldots x_my_m \mid x_i\varepsilon\Delta^*, y_i\varepsilon\Delta^*, x_1x_2\ldots x_m\varepsilon|E|, y_1y_2\ldots y_m\varepsilon|F|\}$

(9) $|E^{\textcircled{\scriptsize$\|$}}| = \bigcup_{i=0}^{\infty} |E^{\|i}|$, where $E^{\|0} = \lambda$, $E^{\|i+1} = E^{\|i} \| E$ for $i \geq 0$.

Remark 1: In this paper concurrent expressions are defined as extended regular expressions with the shuffle operator($\|$) and its closure($\textcircled{\scriptsize$\|$}$). Shaw's definition of flow expressions contains an infinite cycle operation(∞), but the infinity operator(∞) was excluded in this paper since its descriptive power is beyound the standard theory of formal languages. The analysis of flow expressions in this paper is essentially based on the theory of formal languages. In this paper

$\underline{R}(\Delta)$ denotes the set of regular expressions over Δ;

$\underline{C}(\Delta)$ denotes the set of concurrent expressions over Δ.

Remark 2: Let E=ab and F=b as an example. Then we have

$|E\|F| = \{abc, acb, cab\}$

$|E^{\textcircled{\scriptsize$\|$}}| = \{\lambda, ab, abab, aabb, \ldots\}$

The shuffle operation "E$\|$F" corresponds to the set of computation sequence by a concurrent process:

cobegin E $\|$ F coend

2.2 Synchronized Concurrent Expressions

Let Σ be a finite set of symbols and Γ be a finite set of synchronization primitives, where $\Sigma\cap\Gamma = \phi$. Then a concurrent expression over Σ and Γ is called a synchronized concurrent expression, and a language over Γ is called a synchronization mechanism. Given a synchronization mechanism K and a concurrent expression E, the language described by E with K-control is given as

$L_K(E) = \{h(x) \mid x\varepsilon|E|, \overline{h}(x)\varepsilon K\}$

where h and $\overline{h}$ are the following homomorphisms:

$$h(a) = \begin{cases} a & (a\varepsilon\ \Sigma) \\ \lambda & (a\varepsilon\ \Gamma) \end{cases} \qquad \overline{h}(a) = \begin{cases} \lambda & (a\varepsilon\ \Sigma) \\ a & (a\varepsilon\ \Gamma) \end{cases}$$

Let K be a synchronization mechanism and $\underline{K}$ be a family of synchronization mechanisms. Then we define a family of languages in terms of synchronization mechanisms as follows:

$\mathcal{L}_K = \{L_K(E) \mid E\varepsilon\underline{C}(\Sigma\cup\Gamma), \Sigma\cap\Gamma = \phi\}$

$\mathcal{L}_{\underline{K}} = \bigcup_{K\varepsilon\underline{K}} \mathcal{L}_K$

where $\underline{C}(\Sigma\cup\Gamma)$ is the set of concurrent expressions over $\Sigma\cup\Gamma$.

Remark 3: In the above definition Σ and Γ correspond to the set of basic processes and synchronization primitives, respectively. It may be interpreted that an element of x of $|E|$ is a computational sequence of a concurrent process denoted by E.The homomorphisms $h(x)$ and $\overline{h}(x)$ correspond to the sequences of basic processes and synchronization primitives in x,respectively. This formalism of characterizing process descriptions and synchronization mechanisms of synchronized concurrent expressions by homomorphisms is a slightly generalized and formalized version of flow expressions introduced by Shaw[1978],in which only a restricted class of synchronization mechanisms are allowed.Our present formalism enables us to deal with synchronization mechanisms independently of process descriptions.In this paper we study the desciptive power of L_K and $L_{\underline{K}}$ with respect to K and $\underline{K}$ using this formalism.

2.3 Semaphore Systems and Synchronization Primitives in Concurrent Expressions

Let $\Gamma_n=\{\sigma_i,\omega_i \mid i=0,1,\dots,n-1\}$,where σ_i and ω_i correspond to the wait and signal operations on the i-th semaphore variable,respectively.Using concurrent expressions we define four semaphore systems over Γ_n in the following way:

(1) Counter semaphore: $C(n)$

$C(n)=|(\sigma_0\omega_0+\sigma_0)^{\textcircled{\|}}\|(\sigma_1\omega_1+\sigma_1)^{\textcircled{\|}}\|\dots\|(\sigma_{n-1}\omega_{n-1}+\sigma_{n-1})^{\textcircled{\|}}|$

$\underline{C}=\{C(n)\mid n\geqq 0\}$,where $C(0)=\{\lambda\}$

(2) [0]-counter semaphore: $C_0(n)$

$C_0(n)=|(\sigma_0\omega_0)^{\textcircled{\|}}\|(\sigma_1\omega_1)^{\textcircled{\|}}\|\dots\|(\sigma_{n-1}\omega_{n-1})^{\textcircled{\|}}|$

$\underline{C}_0=\{C_0(n)\mid n\geqq 0\}$, where $C_0(0)=\{\lambda\}$

(3) Binary semaphore: $B(n)$

$B(n)=|(\sigma_0\omega_0+\sigma_0)^*\|(\sigma_1\omega_1+\sigma_1)^*\|\dots\|(\sigma_{n-1}\omega_{n-1}+\sigma_{n-1})^*|$

$\underline{B}=\{B(n)\mid n\geqq 0\}$,where $B(0)=\{\lambda\}$

(4) [0]-binary semaphore: $B_0(n)$

$B_0(n)=|(\sigma_0\omega_0)^*\|(\sigma_1\omega_1)^*\|\dots\|(\sigma_{n-1}\omega_{n-1})^*|$

$\underline{B}_0(n)=\{B_0(n)\mid n\geqq 0\}$,where $B_0(0)=\{\lambda\}$

Remark 4: From the standpoint of operating systems practice we have the following correspondence.

1) ω_i and σ_i of $C(n)$ and $C_0(n)$ correspond to the following wait(s_i) and signal(s_i),respectively.

wait(s_i): when s>0 do s←s-1

signal(s_i): when true do s←s+1

2) ω_i and σ_i of $B(n)$ correspond to "when s_i=1 do s_i←0" and "when true do s_i←1",respectively.

3) ω_i and σ_i of $B_0(n)$ correspond to "when s_i=1 do s_i←0" and "when s_i=0 do s_i←1",respectively.

In the above definition of the semaphore systems we can observe the interesting fact that the synchronization mechanisms based on counter semaphores and binary semaphores can be defined as a kind of well-structured concurrent expressions in the sense that they are expressed without any complicated nesting structure and they are readable from left to right.These well-structured descriptions of synchronization mechanisms may be interesting and worthwhile for future studies.

Extending the above definitions of semaphore systems we can define the following general synchronization mechanism:

(5) Let $\underline{CE}(n)$ be a family of synchronization mechanisms defined by concurrent expression with n synchronization primitives.Then we denote

$$\underline{CE} = \bigcup_{n=0}^{\infty} \underline{CE}(n)$$

Notice that in this definition of $\underline{CE}$ n synchronization primitives are not restricted to the semaphores. $C(n)$, $C_0(n)$, $B(n)$ and $B_0(n)$ may be considered to be special cases of $\underline{CE}(2n)$.

(6) A concurrent expression controlled by a semaphore system will be called as a semaphore controlled concurrent expression. In the discussions below we are interested in L_K and $L_{\underline{K}}$ when K and $\underline{K}$ belong to the semaphore systems or $\underline{CE}$.

2.4 Some Previous Results

Under our present formalism we can state the results obtained in the study of flow expressions as follows:

T1: For a concurrent expression E $|E|$ is not necessarily context-free but context-sensitive; if E does not contain the shuffle closure operator ($\oplus$), then $|E|$ is regular.

T2: If a concurrent expression E does not contain any shuffle closure ($\oplus$), then $L_{B(n)}(E)$ is regular. [Shaw]

T3: $L_{\underline{B}} = \underline{RE}$, where $\underline{RE}$ is a family of recursively enumerable sets. [Araki]

T4: $L_{\underline{CE}} = L_{C(6)} = L_{C_0(6)} = L_{B(6)} = L_{B_0(6)}$ [Nishitani-Ito]

According to T1 we can say that any synchronization mechanism which belongs $\underline{CE}$ is a context-sensitive control structure; $B(n)$ and $B_0(n)$ are regular control structures for $n \geq 1$; $C(n)$ and $C_0(n)$ are not regular control structures for $n \geq 1$. In this paper we improve and extend T4 using a different simulation technique, which will be expliaed in the following sections.

2.5 Examples

Before moving onto the detailed discussions we give several examples to give an intuitive appeal of synchronized concurrent expressions.

(1) Correspondence between concurrent expressions and programs

[program]	[concurrent expression]
if B then E else F	$E+F$
begin E; F end	EF
while B do E	E^*
cobegin E ∥ F coend	$E \| F$
for i=1 to n do createprocess[E,i]	$E^{\oplus}$
wait(s)	ω_s
signal(s)	σ_s

(2) $L_{B_0(1)}[(\omega a b \sigma) \| (\omega c d \sigma)] = \{abcd, cdab\}$

$L_B[(\sigma S \omega)^{\oplus}] = |S^*|$ assuming that σ and ω does not appear in S.

$L_B[a \omega b \sigma] = \phi$ (In this case the process is deadlocked.)

III. UNIVERSALITY OF SYNCHRONIZED CONCURRENT EXPRESSIONS

Intuitively a semaphore system may be universal in the sense that any synchronization mechanism in operating systems practice can be realized by a semaphore system. Using the above formalism we give a formal proof that four kinds of semaphore systems given in Section II

can simulate each other

can simulate any synchronization mechanism within $\underline{CE}$

and can describe the family of recursively enumerable sets; eventually we reach the universality of synchronized concurrent expressions.

That is, we can prove the following theorem:

Theorem 1

$$L_{\underline{CE}} = L_{\underline{C}} = L_{\underline{C_0}} = L_{\underline{B}} = L_{\underline{B_0}} = \underline{RE}$$

In order to prove Theorem 1 we need some preliminary definitions and results

Definition: c_i $(0 \leq i \leq n-1)$ is the following mapping from $(\Sigma \cup \Gamma_n)^*$ into the set of integers: for $x \varepsilon (\Sigma \cup \Gamma_n)^*$ and $a\ (\Sigma \cup \Gamma_n)$,

$$c_i(\lambda)=0$$

$$c_i(xa)=\begin{cases} c_i(x)+1 & \text{for } a=\sigma_i \\ c_i(x)-1 & \text{for } a=\omega_i \\ c_i(x) & \text{for } a \notin \{\sigma_i,\omega_i\} \end{cases}$$

Proposition: For $x \varepsilon (\Sigma \cup \Gamma_n)$ $\bar{h}(x) \varepsilon C(n)$ if and only if for any $x' \varepsilon \text{prefix}(x)$ and each $i(0 \leq i \leq n-1)$ $c_i(x') \geq 0$.

Definition: Let Δ and Δ' be the finite sets of symbols and f be a mapping from Δ into $\underline{C}(\Delta')$. For a concurrent expression E on Δ f(E) denotes a concurrent expressio on Δ' which is obtained from E by replacing every occurrence of $a(\varepsilon\Delta)$ by f(a). When $|f(x)|$ denotes a single word for $x(\varepsilon\Delta^*)$, f(x) may be treated as a word on Δ', where we assume $f(\lambda)=\lambda$.

Proposition: Let E be a concurrent expression on Δ and let $f:\Delta \to \underline{C}(\Delta' \cup \{[,]\})$ be the following mapping:

$$f(a)=[E_a] \qquad (a \varepsilon \Delta)$$

where [and] are the symbols which do not belong to Δ' and $E_a \varepsilon \underline{C}(\Delta')$.
If the sequence of [and] in $y \varepsilon |f(E)|$ is in the form of $([])^*$, then there exists $x \varepsilon |E|$ such that $y \varepsilon |f(x)|$, and x and y are given as

$$x=a_1 a_2 \ldots a_m \qquad (a_j \varepsilon \Delta)$$

$$y=[y_1][y_2]\ldots[y_m] \qquad ([y_j] \varepsilon |f(a_j)|)$$

Lemma 1:

$$\mathcal{L}_{\underline{CE}(n)} \subseteq \mathcal{L}_{C(n+3)}$$

$$\mathcal{L}_{\underline{CE}(n)} \subseteq \mathcal{L}_{C_0(n+3)}$$

$$\mathcal{L}_{\underline{CE}(n)} \subseteq \mathcal{L}_{B(n+3)}$$

$$\mathcal{L}_{\underline{CE}(n)} \subseteq \mathcal{L}_{B_0(n+3)}$$

This lemma constitute the major part of Theorem 1. In this paper we give the outline of the proof of this lemma, since the complete proof of this lemma is rather lenghty.

Outline of the proof of Lemma 1: Let $\Gamma=\{\gamma_0,\gamma_1,\ldots,\gamma_{n-1}\}$ be a set of synchronization primitives. In order to establish Lemma 1 we construct a concurrent expression E' on $\Sigma \cup \Gamma_{n+3}$ for a concurrent expression E_K on Γ and a concurrent expression E on $\Sigma \cup \Gamma$. For this concurrent expression E' we show

$$L_{|E_K|}(E)=L_{C(n+3)}(E')=L_{C_0(n+3)}(E')=L_{B(n+3)}(E')=L_{B_0(n+3)}(E')$$

Let $]$, $[$, $\underline{]}$ and $\underline{[}$ be $\sigma_{n+1}, \omega_{n+1}, \sigma_{n+2}, \omega_{n+2}$, respectively, for $\Gamma_{n+3}=\{\sigma_0,\omega_0,\sigma_1,\omega_1,\ldots,\sigma_{n+1},\omega_{n+1},\sigma_{n+2},\omega_{n+2}\}$. We define the following mappings:

$$f_1: \ \Sigma \cup \Gamma \to \underline{C}(\Sigma \cup \Gamma_{n+3})$$

$$f_1(a)=[a] \qquad (a \varepsilon \Gamma)$$

$$f_1(\gamma_i)=[\omega_i \sigma_n] \qquad (\gamma_i \varepsilon \Gamma)$$

$$f_2: \ \Gamma \to \underline{C}(\Gamma_{n+3})$$

$$f_2(\gamma_i)=\underline{[}\sigma_i \omega_n \underline{]} \qquad (\gamma_i \varepsilon \Gamma)$$

Then the required concurrent expression E' can be given as follows:

$$E'=]\underline{]}(f_1(E) \| f_2(E_K)\underline{[}[$$

Lemma 1 can be proved by showing that $L_{|E_K|}(E) \subseteq L_{B_0(n+3)}$

and $L_{C(n+3)}(E') \subseteq L_{|E_K|}(E)$

,since we have $L_{B_0(n)}(E'') \subseteq L_{B(n)}(E'') \subseteq L_{C(n)}(E'')$

and $L_{B_0(n)}(E'') \subseteq L_{C_0(n)}(E'') \subseteq L_{C(n)}(E'')$

for any concurrent expression E''.

Proof of Theorem 1

Using Lemma 1 we can easily prove Theorem 1. From Lemma 1 we have $L_{\underline{CE}} \subseteq L_{\underline{C}}$. From the definition of C(n), $\underline{CE}$(n) and $\underline{CE}$, $C(n) \subseteq \underline{CE}(2n) \subseteq \underline{CE}$, hence we have $L_{\underline{C}} \subseteq L_{\underline{CE}}$. Thus we get $L_{\underline{CE}} = L_{\underline{C}}$; we can establish the similar results for $\underline{C}_0$, $\underline{B}$ and $\underline{B}_0$.

Remark 5: The recursive enumerability of Theorem 1 follows from T3,but the proof of T3 by Araki-Tokura is complicated since they use the original Shaw flow expression and the binary semaphore system for simulation.Using the formalism of this paper we can give a neat proof of $L_{\underline{CE}} = \underline{RE}$ in a straightforward manner,simulating the behaviour of 2-counter automaton by a synchronized concurrent expression. Intuitively speaking this can be done by simulating a counter behaviour as a synchronization mechanism

$$|((\gamma\bar{\gamma})^{\oplus}\gamma_0)^*|$$

where γ, $\bar{\gamma}$ and γ_0 correspond to the increment(+1),the decrement(-1) and the zero test of the counter,respectively.According to this proof we can show the following stronger result:

Corollary

$$L_{\underline{CE}}^{reg} = \underline{RE}$$

where $L_{\underline{CE}}^{reg} = \{L_{\underline{CE}}(E) \mid E \in \underline{R}(\Sigma \cup \Gamma), \Sigma \cap \Gamma = \phi\}$ and $\underline{R}(\Sigma \cup \Gamma)$ is the set of regular expressions over $\Sigma \cup \Gamma$.

That is, the regular expressions controlled by $\underline{CE}$ can describe recursively enumerable sets; hence we can say that synchronized regular expressions are universal.

IV. UNIVERSALITY OF SYNCHRONIZED CONCURRENT EXPRESSIONS WITH THREE SYNCHRONIZATION PRIMITIVES

The results on universality of synchronized concurrent expressions can be strenghtened as follows:

Theorem 2

$$L_{\underline{CE}} = L_{C(3)} = L_{C_0(3)} = L_{B(3)} = L_{B_0(3)} = \underline{RE}$$

The proof of this theorem can be shown by showing the following lemmas, and Lemma 3 constitute the major part of Theorem 2.

Lemma 2

$L_{C(n)} \subseteq L_{C_0(n)}$, $L_{C(n)} \subseteq L_{B(n)}$, $L_{C(n)} \subseteq L_{B_0(n)}$ for n=0,1,2,...

Lemma 3

$L_{B_0(n)} \subseteq L_{C(3)}$ for n=0,1,2,...

Outline of the proof of Lemma 2: The proof of Lemma 2 can be given by showing that for any concurrent expression E on $\Sigma \cup \Gamma_n$ we can construct a concurrent expression E' such that

$$L_{C(n)}(E) = L_{C_0(n)}(E') = L_{B(n)}(E') = L_{B_0(n)}(E')$$

Let f_1 be the following mapping:

$$f_1: \Gamma_n \to \Gamma_n;\ f_1(\sigma_i) = \omega_i \text{ for } \sigma_i \in \Gamma_n;\ f_1(\omega_i) = \sigma_i \text{ for } \omega_i \in \Gamma_n$$

Then the required concurrent expression E' can be given as follows:

$$E'=E\|f_1(E_C)$$

where E_C is the concurrent expression corresponding to C(n), that is,

$$E_C=(\sigma_0\omega_0+\sigma_0)^{\textcircled{\|}}\|\ldots\|(\sigma_{n-1}\omega_{n-1}+\sigma_{n-1})^{\textcircled{\|}}.$$

For this E' we can show

$$L_{C(n)}(E)\subseteq L_{B_0(n)}(E')$$

and $L_{C(n)}(E')\subseteq L_{C(n)}(E)$.

From these results we can prove Lemma 2, since for any expression E" we have

$$L_{B_0(n)}(E'')\subseteq L_{B(n)}(E'')\subseteq L_{C(n)}(E'')$$

$$L_{B_0(n)}(E'')\subseteq L_{C_0(n)}(E'')\subseteq L_{C(n)}(E'')$$

Outline of the proof of Lemma 3: The proof of Lemma 3 can be given by showing that for any concurrent expression E on $\Sigma\cup\Gamma_n$ we can construct a concurrent expression E' on $\Sigma\cup\Gamma_3$ such that

$$L_{B_0(n)}(E)=L_{C(3)}(E')$$

The required concurrent expression E' can be given as

$$E'=S(0)]f(E)[W(0)$$

where (1)] and [correspond to σ_0 and ω_0, respectively.

(2) For integers $t(\geqq 0)$ and $i(\geqq 0)$, let $\langle t\rangle_i$ be the value of the i-th position of the binary representation of t, and let $N=2^n-1$. For an integer $t(0\leqq t\leqq N)$ W(t) and S(t) are defined as the following words on Γ_3:

$$W(t)=\omega_1^t\omega_2^{N-t}$$

$$S(t)=\sigma_1^t\sigma_2^{N-t}$$

(3) $f: \Sigma\cup\Gamma_n\rightarrow\underline{C}(\Sigma\cup\Gamma_3)$

$$f(a)=[a] \qquad \text{for } a\in\Sigma$$

$$f(\sigma_i)=([W(\bar{k}_1)S(k_1)]+[W(\bar{k}_2)S(k_2)]+\ldots+[W(\bar{k}_{2^{n-1}})S(k_{2^{n-1}})]) \quad (\sigma_i\in\Gamma_n)$$

$$f(\omega_i)=([W(k_1)S(\bar{k}_1)]+[W(k_2)S(\bar{k}_2)]+\ldots+[W(k_{2^{n-1}})S(\bar{k}_{2^{n-1}})]) \quad (\omega_i\in\Gamma_n)$$

assuming that $k_j(1\leqq j\leqq 2^{n-1})$ is an integer such that $\langle k_j\rangle_i=1$ and $0\leqq k_j\leqq N(=2^n-1)$,

and $\bar{k}_j=k_j-2^i$.

In order to show that $L_{B_0(n)}(E)=L_{C(3)}(E')$ we need to use the following propositions.

Proposition: For $x\in(\Gamma\cup\Sigma_n)^*$

$$\bar{h}(x)\in B_0(n)$$

if and only if $0\leqq c_i(x')\leqq 1$ for any $x'\in\text{prefix}(x)$ and every $i(0\leqq i\leqq n-1)$ and $c_i(\bar{x})=0$ for every $i(0\leqq i\leqq n-1)$.

Proposition: Let

$$y=S(t_0)]f(u_0)[W(t')S(t_1)]f(u_1)[W(t_2')S(t_2)]f(u_2)\ldots[W(t_m')S(t_m)]f(u_m)[W(t_M')$$

where $u_j\in\Sigma^*(0\leqq j\leqq m)$ and $M=m+1$. Then

$$h(y)\in C(3) \text{ if and only if } t_j=t_{j+1}' \text{ for every } j(0\leqq j\leqq m).$$

Using the above results we can prove our main theorem as follows:

Proof of Theorem 2

By Theorem 1 $L_{\underline{CE}}=L_{\underline{B}_0}$; by Lemma 3 $L_{\underline{B}_0}\subseteq L_{C(3)}$; by Lemma 2 $L_{C(3)}\subseteq L_{C_0(3)}$. $L_{C_0(3)}\subseteq L_{\underline{CE}}$ follows from the definitions. Thus we have proved

$$L_{\underline{CE}}=L_{C(3)}=L_{C_0(3)}$$

Similarly we can prove

$$L_{\underline{CE}}=L_{B(3)}=L_{B_0(3)}.$$

This completes the proof of Theorem 2.

V. CONCLUDING REMARKS

We can summarize our results in this paper as follows:

(1) $L_{\underline{CE}}=L_{\underline{C}}=L_{\underline{C}_0}=L_{\underline{B}}=L_{\underline{B}_0}=\underline{RE}$

(2) $L_{\underline{CE}}=L_{\underline{Rg}}=L_{C(3)}=L_{C_0(3)}=L_{B(3)}=L_{B_0(3)}=\underline{RE}$

where Rg is the class of regular expressions defined over Γ_n.(Notice that B and B_0 belong to Rg.)

It remains open if the class of synchronized concurrent expressions with only two synchronization primitives are universal.

The universality of synchronized concuurent expressions implies a rather negative result for analysis of concurrent processes. This leads us to study a more restricted class of synchronized concurrent expressions.We have already mentioned such a class,called a class of synchronized regular expressions.

A regular expression over Σ and Γ is called a synchronized concurrent expression.Given a synchronization mechanism K we define a family of languages as follows:

$$L_K^{reg}=\{L_K(E)\mid E\in\underline{R}(\Sigma\cup\Gamma),\Sigma\cap\Gamma=\phi\}$$

$$L_{\underline{K}}^{reg}=\bigcup_{K\in\underline{K}}L_K^{reg}$$

For the class of synchronized regular expressions we have the following results:

(i) $L_{\underline{CE}}^{reg}=\underline{RE}$

(ii) $L_{\underline{B}}^{reg}=L_{\underline{Rg}}^{reg}=\underline{REG}$ (REG: the family of regular sets)

(iii) $\underline{REG}=L_{\underline{Rg}}^{reg}=L_{\underline{B}_0}^{reg}=L_{\underline{B}}^{reg}\subsetneq L_{\underline{C}}^{reg}\subsetneq L_{\underline{C}_0}^{reg}=\underline{PN}$

where PN is the family of Petri net languages.

(iv) $L_{C(n)}^{reg}\subsetneq L_{C(n+1)}^{reg}$ $(n\geqq 0)$

(v) $L_{\underline{DYCK}}^{reg}=L_{DYCK(2)}^{reg}=\underline{CFL}$

where DYCK(n) is the Dyck synchronization mechanism over Γ_n defined as follows:

Let $G_D=(V_T,V_N,P,S)$

$V_T=\Gamma_n=\{\sigma_1,\omega_1,\sigma_2,\omega_2,\ldots,\sigma_n,\omega_n\}$

$V_N=S$

$P=\{S\to S\sigma_i S\omega_i S\mid 1\leqq i\leqq n\}\cup\{S\to\lambda\}$

The Dyck synchronization mechanism over Γ_n ---DYCK(n)--- can be defined as the language generated by the above grammar G_D.We also define as

$\underline{DYCK}=\{DYCK(n)|n\geq 0\}$

This result establishes a characterization theorem of context-free languages by a class of synchronized regular expressions.

The details of these results on synchronized regular expressions will be given elsewhere.

REFERENCES

(1) A. C. Shaw: Software descriptions with flow expressions,IEEE Trans. Software Engineering,SE-4,no.3(1978)

(2) A. C. Shaw: Software specification languages based on regular expression ,Software Development Tools,Springer-Verlag(1980)

(3) T. Araki and N. Tokura: Unsolvability of the equivalence problems of flow expressions,Transactions of IECE of Japan,J62-D, no.6 (1979) (in Japanese)

(4) T. Araki and N. Tokura: On the descriptive power of flow expressions and event expressions, Transactions of IECE of Japan,J63-D, no.8(1980) (in Japanese)

(5) Y. Nishitani and T. Ito: On the descriptive power of semaphore controlled flow expressions,Convention Record of IECE of Japan(August 1980) (in Japanese)

(6) P. C. Fisher: Turing machines with restricted memory access,Information and Control,vol. 9,no. 4,(1966)

(7) A. N. Habermann: Introduction to Operationg System Design,Science Reseach Associates(1976)

(8) J. E. Hopcroft and J. D. Ullman: Formal Languages and their Relation to Automata Theory, Addison-Wesley Publishing Company(1969)

CONTROL PROBLEMS IN THE FIELD ARTILLERY SYSTEM

STEPHEN S. WOLFF
Ballistic Research Laboratory
Aberdeen Proving Ground, MD.

ABSTRACT

Problems of fire support control are of growing interest and importance to the Field Artillery community. This paper briefly discusses typical problems of resource allocation, decision-making and fire control in an artillery context. Analytic efforts in this area can contribute both to their understanding in the laboratory as well as to their solution in the field.

INTRODUCTION

Computers are being supplied to the Army's Field Artillery forces in massive quantity: the Battalion TACFIRE is a mainframe computer; the CPU of the Battery Computer System is a competent 18-bit mini; the forward observer's "Digital Message Device" is an 8-bit micro with a large complement of RAM (there are 36 ground-based forward observers in an artillery battalion).

For traditional artillery missions (e.g., time-on-target barrage), centralized control of many howitzers by command levels above Battalion is often necessary. But to be effective in new roles -- particularly the engagement of enemy tanks with "smart" projectiles -- requires a degree of responsiveness of the artillery system that has never been achieved even under benign test conditions.

The problem of battlefield communication among widely scattered and frequently moving "shoot and scoot" howitzers and their command elements is thorny indeed; but even if such communication were always error-free, the issues raised by conflicting and only approximate target acquisition reports, widely-shared databanks, and distributed decision-making processes emerge as some of the knottiest currently facing the tactical army [1].

Somehow, one feels, the enormous (and rather ill-used) computing power now available ought to be able to aid the processes involved in controlling the artillery system. This paper presents in a semi-quantitative way some problems that arise in these processes.

A RESOURCE ALLOCATION PROBLEM

In any large-scale battle, the field artillery operates in what is euphemistically called a "target-rich environment"; that is, there are many more targets (say n) than there are artillery resources to engage them. Suppose that there are r units of artillery (guns, sections, batteries, ...). Then we can imagine an (r x n) matrix P whose typical entry p_{ij} is the probability of successfully engaging target j with resource unit i. Attached to target j is a worth or value v_j accruing from its engagement, so that if $V=\mathrm{diag}(v_j)$ then $U = PV$ is a utility matrix and we are faced with a standard resource allocation problem: maximize $\sum_{i=1}^{r} u_{ij}$ where j runs over a permutation of {1,2,...,n}.

This standard problem is however complicated by some non-standard features. The probabilities p_{ij} are presumably computable automatically; but quite apart from the subjectivity of the values $\{v_j\}$, and their consequent strong dependence on who assigns them, they are in even the most speculative future systems only integer ranks. In current systems they are merely priorities chosen from a small set of integers. Moreover, in a rapidly evolving battle the $\{v_j\}$ are time-varying: "now" may be too soon or "five minutes from now" too late. What evaluation should be used for the time-varying utilities?

A DECISION PROBLEM

Suppose a commander must choose one from among a set of alternatives, and wants to make the choice as quickly as possible. Then it is well known that, for any reasonable assumption on the functional relation between decision time and number of alternatives, n, the decision time T is reduced by using a multi-level decision process. In particular, if $n=s^k$ (so that there are k levels with s choices in each), and if decision time is linear in the number of available choices, then T is minimized by taking s=3. This result implies that the typical and frequently-maligned hierarchical command structure - in which each commander controls three subordinate commands - is a pretty good way to get rapid decisions.

But the foregoing result applies only if the time required to propagate intermediate decisions from one stage to the next is negligible. In fact (under the same assumptions as above), if the inter-stage communication time is equal to the within-stage decision time, the optimal value for s is 4. As communication time increases, the optimal value of s increases monotonically towards n: in the limit the decision is made in a single stage.

It is generally supposed that inter-stage communication times will increase during an intense battle - both because of contention for communication channels and the consequent formation of queues at communication nodes, as well as because of intentional jamming and other countermeasures; this supposition is not contradicted by evidence from field experiments. For this reason (and others) it is becoming conventional wisdom that the command structure of the field artillery system must be "dynamically reconfigurable".

However, what about the quality of the decisions produced by this hierarchical structure?

If the information and control structures are identical (a customary but not necessary assumption), and if the state of the system to be controlled is represented by x, then we may describe the information available to the m^{th}-level commander by $y_m=f_m(y_{m-1},x,\tilde{v}_m)$, where by definition $y_0=x$. The presence of x in the argument list allows for the possibility that the m^{th} level commander may have independent information about the system state (e.g., from radar, RPV's, etc.) not available to his subordinates; $\tilde{v}_m$ is a noise or uncertainty.

Now each y_m affects the m^{th}-level decision-maker's version of the utility matrix through the value assessments v_j as well as through the column dimension of P; thus given a model of the state x we can begin to see a control-theoretic formulation of this field artillery problem emerging.

A CONTROL PROBLEM

The fire control prediction problem for artillery-launched "smart" projectiles used against tanks differs from its counterpart for direct-fire antitank guns in two important respects: (1) The projectile time-of-flight T_f (which is the required prediction time) is for direct-fire weapons about 1 - 2 seconds; because of the higher trajectory and greater range, T_f for artillery rounds varies from about 20 seconds to as much as a minute. (2) A direct-fire antitank projectile has to hit its target to inflict any damage: the "physical" and "effective" target sizes are equal; typical "smart bullets" have an acquisition basket or "footprint" of some dozens of metres in radius: the effective target size is much greater than the physical size because of the bullet's terminal homing capability.

Current doctrine for the use of artillery-launched smart bullets assumes away the most difficult part of the lead prediction problem by preselecting projectile impact points at places where targets must eventually be (bridges, narrow roads, etc.), and back-estimating the projectile's launch epoch from an estimate of target velocity obtained by two measurements of target position separated by a known time interval. But if, as now seems likely, the fraction of smart bullets in the artillery's inventory grows, they will ultimately come to be used when targets are not so constrained - i.e., when they are free to move over (relatively) featureless terrain. Under such circumstances, the simple "first-order" lead predictor described above is notoriously inaccurate even at prediction times of a couple of seconds.

We have accordingly looked at the prediction problem from a different standpoint with encouraging early results. Our formulation lacks a certain degree of realism, but it nevertheless holds promise for the future.

Suppose that an artillery gun is assigned to defend a region D of the plane against encroachment by a tank initially located at a point x exterior to D. We assume that the gun can know the exact location of the tank at every instant. Every T seconds, the gun aims and fires a projectile having the property that it destroys the tank if, when the projectile lands T seconds after its launch, the tank is located within a semi-open square of side 2r centered on the projectile's impact point. If the tank is not destroyed, the gun immediately re-aims and fires again; in the T seconds before the new projectile arrives, the tank is free to move anywhere within a square of side 2R (R=2nr, n an integer). We look on this sequence of activities as a two-person zero sum game: the tank "wins" if it penetrates the protected region D and is not destroyed by the next projectile; the gun wins otherwise. This game is a two-dimensional, dynamical form of a somewhat modified "Bomber-Submarine Problem" known for many years [2]; a one-dimensional version has been discussed by Kumar and Shiau [3].

We have shown that the game has a value in randomized strategies, and we have produced a usable algorithm for calculating the value and the optimal strategies. An example is shown in figure 1 for the case that D is the half-plane $x<0$; the ordinate j is the probability of a win by the tank starting from a initial position $x_0=2mr$, the curves are parametrized by n = R/2r: the ratio of the half-side of the tank's "mobility square" to the side of the projectile's "destruction square".

CONCLUSION

This brief article has touched on only a few of the modeling and analysis problems generated by the availability of new weapons, new computers, and new means of communication to the Field Artillery. Many more remain to be defined, explicated and solved.

REFERENCES

[1] Reichard, B., "Fire Support Control at the Fighting Level", Field Artillery Journal, Vol. 49 no. 2, Mar-Apr, 1981; also in Proceedings of the Fourth MIT/ONR C^3 Workshop (San Diego, CA, June 1981) to be published.

[2] Karlin, S., "Mathematical Methods and Theory in Games, Programming, and Economics", Vol II, Addison-Wesley, 1959 (p. 34)

[3] Kumar, P. R. and T. H. Shiau, "Zero-Sum Dynamic Games", in "Advances in Control Theory", C. T. Leondes, ed., Academic Press, 1981 (pp. 345-378)

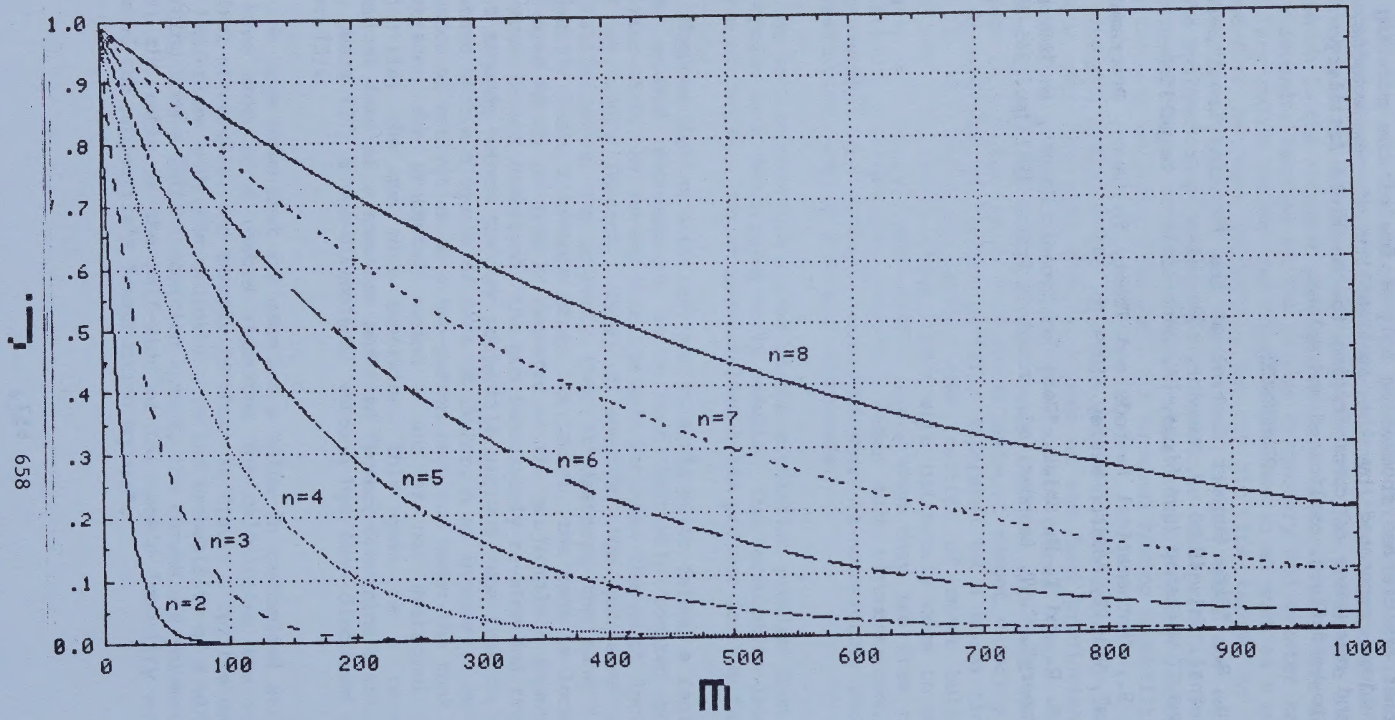


Figure 1

Probability of a Tank Win vs. Starting Position

MICROPROCESSOR-BASED OPTIMAL CONTROLLERS FOR A HELICOPTER TURRET CONTROL SYSTEM

N. COLEMAN
U.S. Army Armament Reseach and Development Command
DRDAR - SCF - CC, Dover, New Jersey 07801

N. K. LOH
Center for Robotics and Advanced Automation
School of Engineering, Oakland University, Rochester, MI 48063

ABSTRACT

The design and implementation of microprocessor-based discrete-time optimal controllers for the XM-97 helicopter gun-turret control system is considered. Nonfiring and firing test results are presented. Significant performance improvement of the optimal turret over the original turret has been obtained in terms of percent overshoot, settling time, and shot pattern dispersion.

1. INTRODUCTION

The design of a high precision helicopter pointing control system using modern control and observer theory has been investigated by Coleman, Loh, et al in [1]. The pointing control system investigated was the XM-97 helicopter gun-turret control system. The continuous-time optimal controllers and observers developed in [1] were implemented by using standard analog electronic components. The resulting performance improvement of the optimal turret control system, in terms of round dispersion, turret overshoot, turret settling time, etc., ranged from a factor of 2 to 1 to a factor 10 to 1 when compared with the performance of the original system.

Recently, the continuing design efforts on helicopter turret control systems have been directed towards the development of high precision turrets employing microprocessor-based optimal controllers. Results presented in [2] demonstrated that it is indeed feasible to implement such an optimal controller in a real time environment by using state of the art microprocessors. Furthermore, numerical processor chips such as the 8087 are sufficiently fast enough to permit implementing some forms of adaptive control laws which may further enhance control system performance.

The purpose of this paper is to present some results on the performance of the XM-97 helicopter turret investigated in [1] employing microprocessor-based optimal controllers. The microprocessor system used was a basic Intel 220 development system expanded to 64K bytes RAM and 1.25 megabytes disc.

2. DESCRIPTION OF THE EXISTING XM-97 HELICOPTER GUN-TURRET CONTROL SYSTEM

The XM-97 helicopter gun-turret system consists of a three-barrel 20mm automatic cannon system and a turret drive system mounted under the nose section of a Cobra helicopter. For test purposes, the gun-turret and helicopter airframe are suspended from a six-degree-of-freedom simulator as shown in Fig. 1. The gun-turret control system is essentially an inertial load driven by a pulse width modulated split seried DC motor through a compliant gear box. The transfer functions of

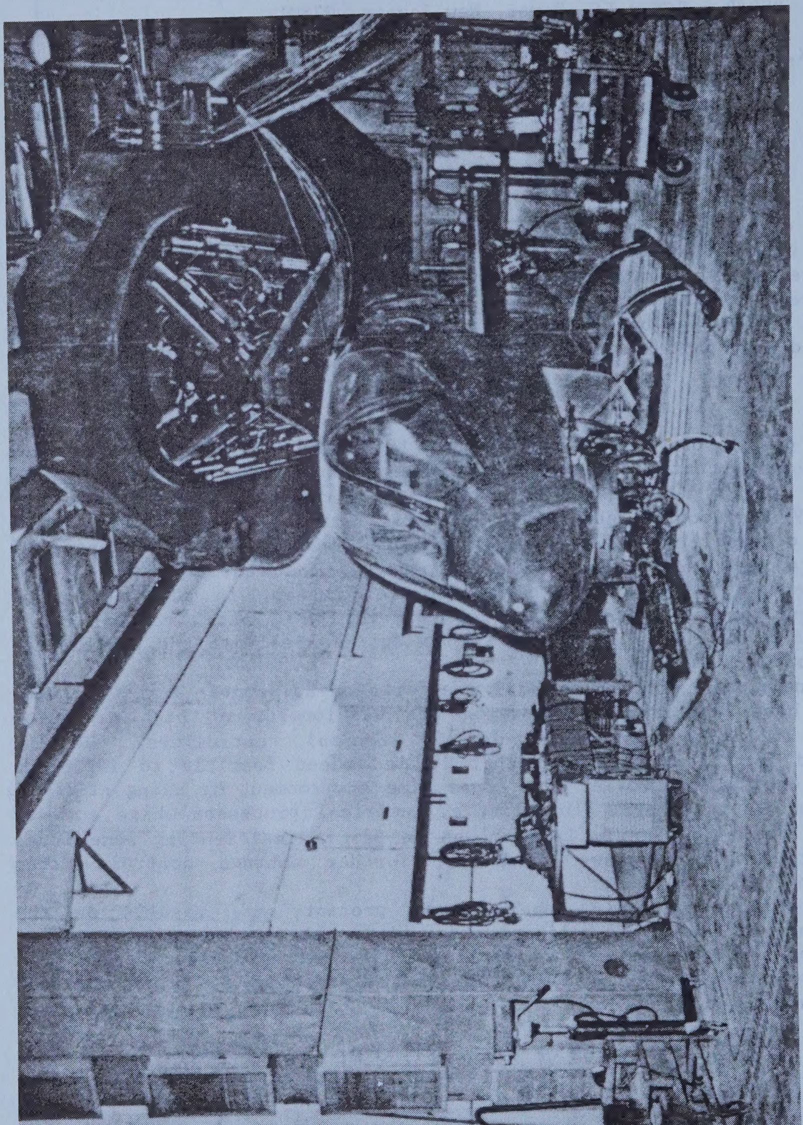


Fig. 1 XM-97 HELICOPTER GUN-TURRET SUSPENDED FROM SIMULATOR

the system are as shown in Fig. 2. The system consists of two controllers: one controller positions the gun turret in azimuth and the other elevates and depresses the gun cradle and the gun. The two controllers are functionally similar and independent. As shown in Fig. 2, the only difference between the two controllers is the gear ratio N which is $N = 620$ for the azimuth channel and $N = 810$ for the elevation channel.

The existing gun-turret control system essentially employs angular position feedback and angular velocity feedback. With the state variables chosen as shown in Fig. 2, the turret dynamics is described by the following 8-dimensional vector differential equation (for both azimuth and elevation channels),

$$\dot{\underline{x}}(t) = A\underline{x}(t) + Bu(t), \ \underline{x}(0) = \underline{x}_o, \tag{1}$$

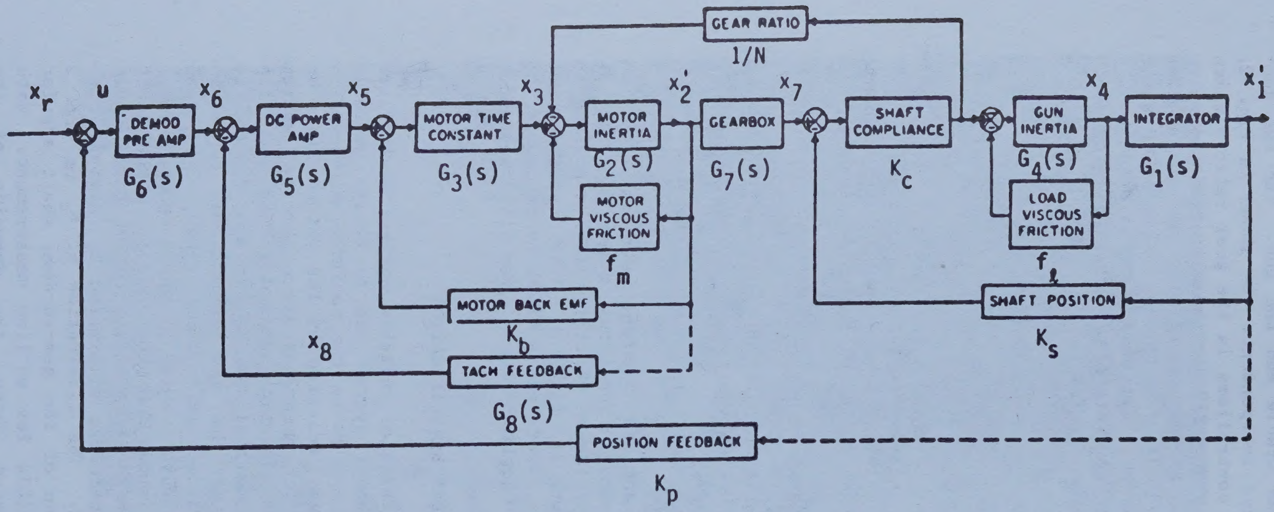
where

$\underline{x}(t) = [x'_1(t)\ x'_2(t)\ x_3(t)\ x_4(t)\ x_5(t)\ x_6(t)\ x_7(t)\ x_8(t)]^T$,
$x'_1(t)$ = gun turret angular position relative to the hull (radians),
$x_4(t)$ = gun-turret angular velocity relative to the hull (radians/second),
$x'_2(t)$ = motor angular velocity relative to the hull (radians/second),
$x_3(t)$ = motor torque (foot-pounds),
$x_5(t)$ = power amplifier output (volts),
$x_6(t)$ = low level electronics output (volts),
$x_7(t)$ = geared-down shaft angular position relative to the hull (radians),
$x_8(t)$ = output of tachometer feedback loop (volts),
$x_r(t)$ = gunner command input (radians),
$u(t)$ = control input (volts) = $x_r(t) - x'_1(t)$,

and A and B are, respectively, 8x8 and 8x1 constant matrices which can easily be determined from the transfer functions given in Fig. 2. Typical step response of the existing turret control system described by (1) for $x_r(t) = 2^o$ is as shown in Fig. 3. The response was oscillatory. For the azimuth channel, there was a 48.5% overshoot in the gun angular position $x'_1(t)$ and the settling time t_s for $x'_1(t)$ was about 0.5 second.

3. GUN-TURRET MODEL SIMPLIFICATION

The first step involved in the design of suitable optimal controllers for the XM-97 helicopter turret control system was to obtain suitable open-loop models for the turret system. Hence the position and velocity feedback paths in the existing system indicated by the dotted lines in Fig. 2 were first removed; a 7th order open-loop turret control system resulted for each of the azimuth and elevation channels. The next step was to simplify the 7th order mathematical model. It was decided to ignore stable poles of magnitudes larger than 150 corresponding to a time constant of $\tau = 1/150 = 6.67$ milliseconds. Hence $G_3(s)$, $G_5(s)$ and $G_6(s)$ in Fig. 2 become, respectively, $G_3(s) \simeq p_1$, $G_5(s) \simeq 7.5$, $G_6(s) \simeq 535$. Furthermore, it was found that the effect of the shaft compliance K_c on the response of the helicopter gun-turret drive could be ignored. The parameter K_c was therefore eliminated by setting it to an arbitrary large value in Fig. 2. The elimination of K_c in Fig. 2 has also resulted in the elimination of the geared-down shaft angular position $x_7(t)$ which is not accessible for on-line measurement. With all the simplifications as discussed above, the dynmaics of the simplified gun-turret can be shown to be described by (see Fig. 4)


x_r
u
DEMOD PRE AMP
$G_6(s)$
x_6
DC POWER AMP
$G_5(s)$
x_5
MOTOR TIME CONSTANT
$G_3(s)$
x_3
MOTOR INERTIA
$G_2(s)$
MOTOR VISCOUS FRICTION
f_m
x_2'
GEARBOX
$G_7(s)$
x_7
GEAR RATIO
1/N
SHAFT COMPLIANCE
K_c
GUN INERTIA
$G_4(s)$
LOAD VISCOUS FRICTION
f_ℓ
x_4
INTEGRATOR
$G_1(s)$
x_1'
MOTOR BACK EMF
K_b
TACH FEEDBACK
$G_8(s)$
x_8
SHAFT POSITION
K_s
POSITION FEEDBACK
K_p

$G_1(s) = \frac{1}{s}$ $\quad$ $G_2(s) = \frac{1}{0.00025s}$ $\quad$ $G_3(s) = \frac{0.02}{1 + 0.002s}$ $\quad$ $G_4(s) = \frac{1}{15.7s}$

$G_5(s) = \frac{7.5}{1 + 0.002s}$ $\quad$ $G_6(s) = \frac{535}{1 + 0.005885s}$ $\quad$ $G_7(s) = \frac{1}{Ns}$ $\quad$ $G_8(s) = \frac{p_1 s}{s + p_2}$

$K_p = K_s = 1.0$ $\quad$ $K_b = 0.0192$ $\quad$ $K_c = 5000000.0$ $\quad$ $f_m = 0.0$ $\quad$ $f_\ell = 50.0$

$$N = \begin{cases} 620 \text{ (AZIMUTH)} \\ 810 \text{ (ELEVATION)} \end{cases} \qquad p_1 = \begin{cases} 0.03 \text{ (AZIMUTH)} \\ 0.375 \text{ (ELEVATION)} \end{cases} \qquad p_2 = \begin{cases} 7.0 \text{ (AZIMUTH)} \\ 6.0 \text{ (ELEVATION)} \end{cases}$$

Fig. 2 ORIGINAL XM-97 HELICOPTER TURRET CONTROL SYSTEMS - AZIMUTH AND ELEVATION CHANNELS

$$\begin{bmatrix} \dot{x}'(t) \\ \\ \dot{x}_2'(t) \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{N} \\ \\ 0 & -\frac{3.84\text{x}10^3}{J} \end{bmatrix} \begin{bmatrix} x'(t) \\ \\ x_2'(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \\ \frac{80.25}{J} \end{bmatrix} u(t), \qquad (2)$$

where $x'_1(t)$ and $x'_2(t)$ are the same as in (1), $J = 3\text{x}10^{-4}$ for the azimuth channel and $J = 2.7\text{x}10^{-4}$ for the elevation channel.

4. DESIGN OF DISCRETE-TIME OPTIMAL TURRET CONTROLLER

The design of continuous-time optimal controllers for the XM-97 helicopter gun-turret control system has been investigated in [1] and [3]. In [1] and [3], the design problem was formulated as shown in Fig. 4, with the dynamics of the open-loop system described by, for both the azimuth and elevation channels,

$$\dot{\underline{x}}(t) = A\underline{x}(t) + Bu(t) + Fv_r, \ \underline{x}(0) = \underline{x}_o, \qquad (3)$$

where

$\underline{x}^T(t) = [x_1(t) \ \ x_2(t)]$,
$x_1(t) = x_r(t) - x'_1(t)$
$=$ error between the position common input x_r (radians) and the actual gun-turret angular position $x_1'(t)$ (radians),
$x_2(t) = Nv_r - x'_2(t)$
$=$ error between the velocity command v_r (radians/second) and the actual motor angular velocity $x_2'(t)$ (radians/second),
$x_r(t) = x_r + v_r t$
$=$ step-plus-ramp position command input (radians),
$u(t) =$ control input (volts),

and A, B and F are constant matrices given by

$$A = \begin{bmatrix} 0 & \frac{1}{N} \\ \\ 0 & \frac{3.84\text{x}10^3}{J} \end{bmatrix} \triangleq \begin{bmatrix} 0 & a_{12} \\ \\ 0 & a_{22} \end{bmatrix},$$

$$B = \begin{bmatrix} 0 \\ \\ \frac{-80.25}{J} \end{bmatrix} \triangleq \begin{bmatrix} 0 \\ \\ -b_{12} \end{bmatrix}, \quad F = \begin{bmatrix} 0 \\ \\ \frac{3.84\text{x}10^{-3}N}{J} \end{bmatrix} \triangleq \begin{bmatrix} 0 \\ \\ f_{12} \end{bmatrix}.$$

To design suitable microprocessor-based optimal controllers for the helicopter gun-turret control system, equation (3) is first discretized by using a sampling interval T seconds as follows:

$$\underline{x}(k+1) = A_d\underline{x}(k) + B_d u(k) + F_d v_r,\ \underline{x}(0) = \underline{x}_o, \tag{4a}$$

where $\underline{x}(k) = \underline{x}(kT)$, $u(k) = u(kT)$,

$$A_d \overset{\Delta}{=} e^{AT} = \begin{bmatrix} 1 & a_{d12} \\ 0 & a_{d22} \end{bmatrix}, \tag{4b}$$

$$B_d \overset{\Delta}{=} \int_o^T e^{At} Bdt = \begin{bmatrix} b_{d11} \\ b_{d12} \end{bmatrix}, \tag{4c}$$

$$F_d \overset{\Delta}{=} \int_o^T e^{At} Fdt = \begin{bmatrix} f_{d11} \\ f_{d12} \end{bmatrix}, \tag{4d}$$

and the components in the A_d, B_d and F_d matrices can be computed easily for each of the azimuth and elevation channels.

The design objective is to drive $\underline{x}(k)$ to the zero state and in the same time minimizing a quadratic performance measure. To achieve the objective, the control $u(k)$ is first split into two parts as

$$u(k) = u_{fb}(k) + u_{ff}(k), \tag{5}$$

where $u_{fb}(k)$ is the feedback component responsible for driving the state $\underline{x}(k)$ to the zero state, and $u_{ff}(k)$ is the feedforward component responsible for accommodating the velocity command v_r, i.e.,

$$B_d u_{ff}(k) + F_d v_r = 0. \tag{6}$$

Using (3), (4) and (6), it is not difficult to show that

$$u_{ff}(k) = \frac{f_{12}}{b_{12}} v_r = 4.785N\times10^{-5} v_r = k_r v_r. \tag{7}$$

Substituting (5) and (7) into (4) yields

$$\underline{x}(k+1) = A_d\underline{x}(k) + B_d u_{fb}(k),\ \underline{x}(0) = \underline{x}_o. \tag{8}$$

Consider the performance measure

$$J_d = \sum_{k=0}^{\infty} [x^T(k)Q_d\underline{x}(k) + 2\underline{x}^T(k)S_d\underline{u}_{fb}(k) + \underline{u}_{fb}^T(k)R_d\underline{u}_{fb}(k)], \tag{9}$$

where Q_d is an nxn (n=2 for the present problem) symmetric positive-semidefinite weighting matrix, R_d is an rxr (r=1 for the present problem) symmetric positive-definite weighting matrix, and S_d is an nxr (n=2, r=1) weighting matrix. In the design of an optimal control system, the numerical values of the elements of the weighting matrices Q_d, R_d and S_d are often chosen in a trial and error basis, guided by, perhaps, the physics of the problem and design experience.

The optimal control which minimizes the performance measure J_d given by (9) subject to (8) is given by [5]

$$u_{fb}(k) = -(R_d + B_d^T K_d B_d)^{-1}(B_d^T K_d A_d + S_d^T)\underline{x}(k) \tag{10a}$$

$$= k_1 x_1(k) + k_2 x_2(k) \tag{10b}$$

$$= k_1[x_r(k) - x_1'(k)] + k_2[Nv_r - x_2'(k)], \tag{10c}$$

where k_1 and k_2 are constants, and K_d is the symmetric positive-definite solution of the algebraic matrix Riccati equation

$$K_d = A_d^T K_d A_d + Q_d$$
$$-(B_d^T K_d A_d + S_d^T)^T (R_d + B_d^T K_d B_d)^{-1} (B_d^T K_d A_d + S_d^T). \tag{11}$$

Combing (7) and (10), the total control is given by

$$u(k) = -(R_d + B_d^T K_d B_d)^{-1}(B_d^T K_d A_d + S_d^T)\underline{x}(k) + k_r v_r \tag{12a}$$

$$= k_1 x_1(k) + (k_2 x_2(k) + k_r v_r. \tag{12b}$$

5. TEST RESULTS

Firing and nonfiring tests of the microprocessor controlled XM-97 helicopter turret control system were conducted using different values for Q_d, R_d and S_d, and two sampling intervals T=1ms and T=10ms. It was found that the case of T=1ms gave the best results.

(a) Nonfiring Tests

For nonfiring tests, typical step responses of the azimuth channel of the original turret and the optimal digital turret are as shown in Figs. 3 and 5. The step input used was $x_r(t)=2^\circ$ for both the original and optimal turrets.

The statistics of the step responses shown in Fig. 3 and 5 are summarized in Table 1. The step responses of the original turret exhibited an average overshoot of 49% while there was no overshoot for the optimal digital turret. The settling time of the original turret was about 3.6 times longer than that of the optimal turret.

TABLE 1: STEP RESPONSES OF ORIGINAL AND OPTIMAL TURRETS-AZIMUTH CHANNEL

TURRET	SETTLING TIME (SEC)	% OVERSHOOT	COMMENTS
Original 2^{o} Step	0.50	49%	Average of left & right excursions
Optimal T=1 ms	0.14	None	Smooth response

(b) Firing Tests

The firing tests were conducted using 20 rounds burst fire. The firing rate was 600 rounds per minute. In order to ignore the transient response of the turret, the statistics of the first 6 rounds, corresponding to approximately 0.6 seconds of firing time[1], were neglected. Typical performances of the original turret and the microprocessor controlled optimal turret in terms of the standard deviations of their round dispersions are given in Table 2. Significant performance improvement was obtained. Further tests using disturbance cancellating optimal digital controllers are currently underway. The results will be reported elsewhere.

TABLE 2: STANDARD DEVIATIONS FOR RANGE OF 85 FEET

TEST NO.	OVERALL AZIMUTH (INCH)	BARREL 1 AZIMUTH (INCH)	BARREL 2 AZIMUTH (INCH)	BARREL 3 AZIMUTH (INCH)
D 30	1.25	0.70	1.15	76
D 31	1.78	1.17	0.80	2.42
O 33	2.11	1.35	2.23	0.99
D 36	1.57	1.47	1.46	1.53
D 37	1.87	1.70	1.14	1.38
O 38	2.26	2.28	1.93	1.80

D = Optimal Digital Turret
O = Original Turret

---

[1] The settling time of the original turret for nonfiring test was about 0.5 second; see Table 1.

KNOWLEDGEMENT

The authors would like to thank Mr. R. Radkiewicz and Mr. J. Strahl of the Ware Simulation Center, Rock Island Arsenal, Rock Island, Illinois for conducting the firing and nonfiring tests and providing the data presented in this paper.

REFERENCES

[1] N. Coleman, K. Lee, N. K. Loh and D. H. Chyung, "Application of Modern Control Theory to the Design of a Helicopter Turret Control System," Proceedings of Second Meeting of the Coordinating Group on Modern Control Theory, Army Material Systems Analysis Activity, Aberdeen Proving Ground, Maryland 21005, Part 1, pp. 1-20, Dec. 1980.

[2] K. Lee, E. Carroll, R. Johnson and B. Kullack, "Microcomputer Implementation of Control Algorithms for Weapon Stabilization and Control," Proceeding of Army Numerical Analysis Conference, Feb. 1981.

[3] N. K. Loh and K. C. Cheok, "Design of Optimal Controllers for XM-97 Helicopter Turret Control System," Final Report for Contract DAAA09-77-C-2040, Technical Report TR-80-11-160, School of Engineering, Oakland University, Rochester, Michigan 48063.

[4] B. D. O. Anderson and J. B. Moore, Linear Optimal Control, Englewood Cliffs, New Jersey: Prentice-Hall, 1971.

[5] B. C. Kuo, Digital Control Systems, New York: Holt, Rinehart and Winston, 1980.

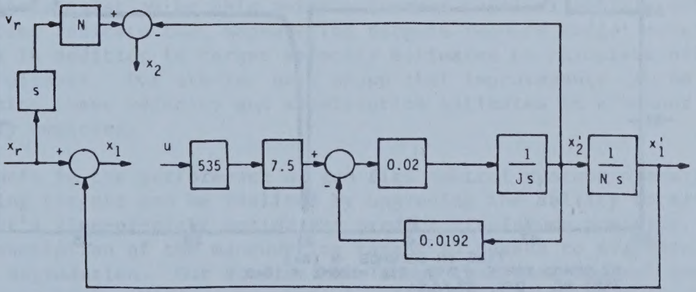


Fig. 4 SIMPLIFIED OPEN-LOOP CONTROL SYSTEM FOR GUN-TURRET

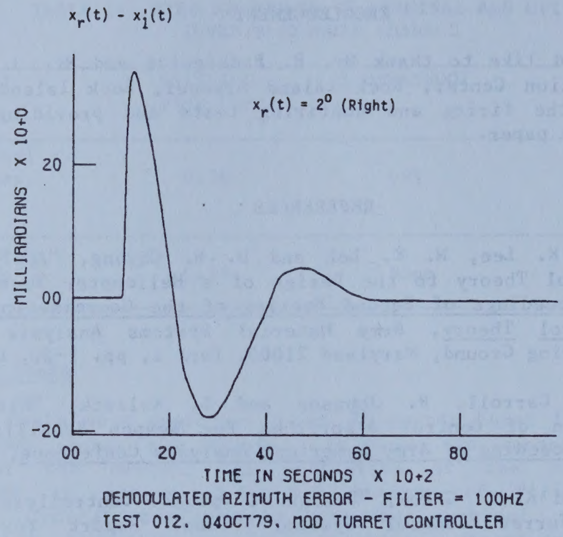


Fig. 3 STEP RESPONSE OF ORIGINAL TURRET

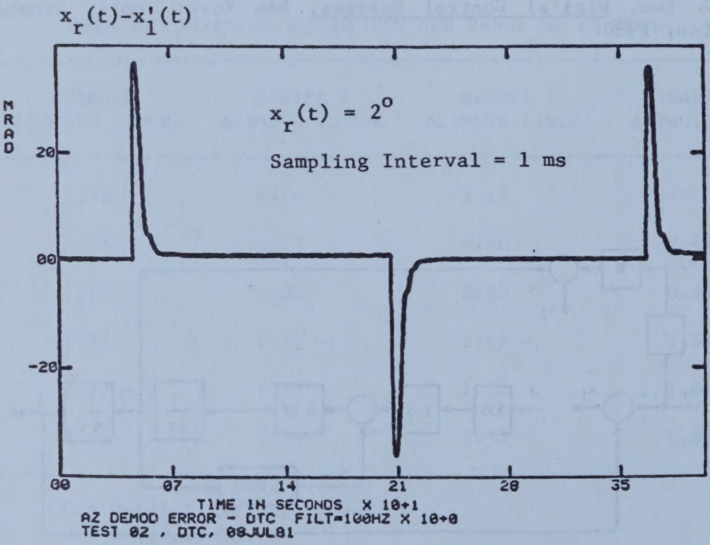


Fig. 5 STEP RESPONSE OF OPTIMAL TURRET

CONCEPTS FOR IMPROVED GUN FIRE CONTROL SYSTEMS

HAROLD H. BURKE
US Army Materiel Systems Analysis Activity
Aberdeen Proving Ground, MD 21005

ABSTRACT

A gun fire control prediction algorithm is presented which will increase performance when maneuvering targets are engaged. Two variations of the concept are discussed; a one dimensional discrete solution and a two dimensional continuous solution. In both versions, target motion is estimated relative to a coordinate frame aligned along the body axes, where velocities and accelerations originate, then transformed to the line-of-sight (LOS) coordinate frame to determine gun line offsets. Lower bound errors for second order and curvilinear predictions are defined. Target maneuvering characteristics dictate which version to adopt. Gun-line servo performance requirements can be reduced when the one dimensional discrete solution is applicable.

INTRODUCTION

The function of a gun fire control system is to offset the gun line from the target line-of-sight, causing a projectile to intercept the target a time-of-flight after firing the gun. Two classifications of target motion exist; non-maneuvering and maneuvering. The projectile-target closest approach is a measure of miss distance. Maneuvering targets require an offset having a magnitude related to the non-maneuvering target offset plus an additional amount related to the target acceleration, acceleration rate, etc., combined with the appropriate functions of projectile time-of-flight. The most familiar type fire control system uses target velocity or target velocity and acceleration combined with time-of-flight to determine gun line offset. Application of sub-optimal estimation methodology, has resulted in the development of velocity plus acceleration controlled offsets that has increased the performance of gun fire control systems against maneuvering targets. The penalty associated with the adoption of these types of systems is the generation of noise associated with the estimation of acceleration. It is relatively simple to estimate "good" velocities, but such is not the case for accelerations. Miss distance is penalized by errors introduced in the offset calculation by this noise. Current tactical vehicle mobility capabilities indicate that maneuvering targets require target acceleration estimates in addition to target velocity estimates to calculate effective gun line offsets. Our studies have shown that improvements can be realized by utilizing these velocity and acceleration estimates in a manner not previously employed.

Improvements in the performance of gun fire control systems operating against maneuvering targets can be realized by upgrading the ability to estimate the target's line-of-sight motion and predict its future position. A quantitative description of the maneuvering target is needed to evaluate the performance degradation. Our studies have shown that fire control performance improvement may be obtained by utilizing fundamental physical laws to characterize the nature of maneuvering targets. Acceleration occurs from either a change in magnitude or direction of velocity or both. Any small portion of a maneuvering path may be considered as an arc of a circle having a fixed radius. The magnitude of this radius may change at different locations on the path, thereby producing a meandering motion. Land, sea and air vehicles have the inherent ability to maneuver on such meandering paths. The maneuvering level of a particular vehicle is constrained by design, but the orientation of its velocity and acceleration is closely related to the geometric configuration. Laws of aerodynamics, hydrodynamics and the requirement to maintain a stable operating condition constrain the orientation of acceleration to be in a direction which is essentially perpendicular to the longitudinal axis of the maneuvering vehicle, which is nominally aligned with its velocity vector. In a similar manner, steering mechanisms for land vehicles employing Ackermann, Fifth Wheel Pivoting or Track Steering, develop the majority of their acceleration by changing the direction of the velocity when operating in a stable condition. The acceleration of maneuvering targets in the atmosphere or on sea or land is therefore, a major function of rotational motion and to a lesser extent translational motion.

GUN FIRE CONTROL SYSTEMS

The purpose of this section is to define the nature of the gun fire control problem and to understand the performance capabilities of a specific gun fire control system designed to solve the problem. The end result of the gun fire control system's solution is to have a projectile that has been previously fired impact the target that was sighted on a time of flight earlier. Two classes of target conditions are possible; stationary and moving. Moving targets can be further divided into maneuvering and non-maneuvering targets. Non-maneuvering targets are characterized by constant speed and heading motion and maneuvering targets are characterized by non-constant speed and/or heading motion. The distance between the target and the engaging fire control system determines the projectile time of flight.

Error in the ability of a fire control system to cause a projectile to intercept the target a time of flight later is referred to as total gun pointing (TGP) error. The TGP error is made up of the errors occurring in the system, or system induced (SI) errors and target induced (TI) errors, induced by target motion during the time of flight of the projectile to the target. A detailed definition of these errors and the parameters they are related to is:

> TGP ERROR : Error in gun pointing is defined as the difference between the actual gun pointing direction (at round exit) and the orientation of the target centroid at time of round impact. The ideal gun orientation is the direction which the fire control system must launch a projectile to hit the target centroid.
>
> SI ERROR : Error in gun pointing caused by sight pointing not being in coincidence with the true LOS to the target at the time of firing plus the difference between estimates of the LOS movement and the true LOS movement, at the time of firing, propagated through a projectile time of flight. All other system-induced errors are not considered in this study.
>
> TI ERROR: Error in gun pointing caused by the target maneuvering during projectile flight time. It is dependent on the order of prediction process of the fire control system. For the first order lead system, the TI Error is the difference between the actual LOS movement during a projectile-time-offlight and the propagated LOS movement, assuming perfect LOS rate, at the time of firing. For a second order lead system, the TI ERROR is the difference between the actual LOS movement during a projectile-time-of-flight and the propagated LOS movement, assuming perfect LOS rate and perfect LOS acceleration, at the time of firing. For a predictor order higher than second order, knowledge of the change in LOS acceleration during the flight time of the projectile has the potential to further reduce the TI error.

ONE DIMENSIONAL DISCRETE CURVILINEAR SOLUTION

Performance when engaging maneuvering targets is significantly degraded relative to performance when non-maneuvering targets are engaged because of an inability to continuously develop the required offsets. For situations where the movement of the maneuvering vehicle is small compared to the range between the maneuvering target and the engaging fire control system, some of this loss in performance may be recovered by incorporating the concepts enumerated in this section. Modifications that alter fire control operation, permitting performance improvement are: (1) detect discrete times when the target's motion is most favorable for obtaining an improved one dimensional fire control solution,i.e.: when the LOS rate or the presented area of the target is minimized (2) actuate the gun offset servos only during time periods that are related to these discrete periods (3) incorporate a modified parabolic prediction algorithm that attributes curvilinear motion to maneuvering targets that are closely related to movements observed from empirical data. The requirement to sense and estimate target velocity and acceleration is fundamental to the principle of operation of the discrete system, but utilization of these estimates in providing gun offsets is considerably different than their use in conventional fire control systems.

Further improvement in the closest approach miss distance may be realizable by modifying the form of an acceleration or conventional parabolic predictor to approximate a circular predictor. If the velocity of the vehicle is constant through an arc, the apparent position, velocity and acceleration perpendicular to the LOS exhibits simple harmonic motion. Where the LOS reverses direction, the apparent velocity crosses zero and the apparent acceleration is at a maximum.

How closely a second order or parabolic predictor approximates the apparent motion of this maneuver in the vicinity of LOS reversal will be discussed. Apparent motion about t_o on Figure 1 is described by

$$\Delta y = -a + a\cos \omega t$$

Representing $\cos\omega t$ in a series

$$\Delta y = -a + \left[a - \frac{a(\omega t)^2}{2!} + \frac{a(\omega t)^4}{4!} - \ldots\right]$$

$a\omega^2$ is the magnitude of acceleration normal to the velocity or the apparent acceleration at t_o and is redefined as A_N. Then

$$\Delta y = -A_N \frac{t^2}{2}\left[1 - \frac{\omega^2 t^2}{12}\right]$$

the apparent motion in the vicinity of LOS reversal for a parabolic predictor is

$$\Delta y = -A_N \frac{t^2}{2}$$

The $\omega^2 t^2/12$ term represents the one dimensional apparent motion error between circular and parabolic predictors. Comparison of the ability of a parabolic and circular predictor to duplicate the apparent displacement resulting from simple harmonic motion is shown in Figure 1.

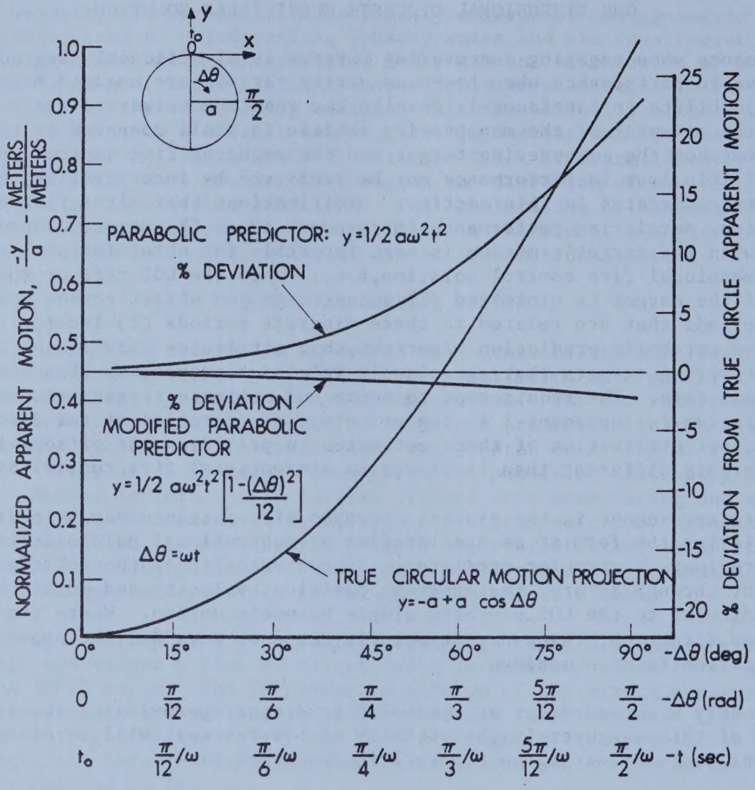


Figure 1. Performance of Parabolic and Modified Parabolic Predictors Compared to True Circular Motion.

The angle $\Delta\Theta$ is measured from t_o and is the product of the angular frequency ω and projectile time-of-flight t. The apparent displacement is a projection of a. For a $\Delta\Theta$ arc of 45° ($\pi/4$ rad.) the parabolic predictor is 5% greater than the projection of movement on the exact circle and the curvilinear predictor is 0.34% less. Similar values for $\Delta\Theta$ arcs of 60° and 90° are 9.6% and 23% greater and 0.8% and 2.2% less, respectively. The radius of curvature combined with the vehicle's tangential velocity determines the apparent acceleration at t_o. The magnitude of the $\Delta\Theta$ arc is a linear function of vehicle velocity, radius of curvature and projectile time-of-flight. Dependent upon the relationship between these parameters the magnitudes of $\Delta\Theta$ and a are established. Maneuvering land vehicles and rotary winged aircraft require offset arcs in the vicinity of 60° when projectile flight times of approximately 2 sec. are considered while maneuvering fixed wing air vehicles require offset arcs in the vicinity of 15° to 30° for similar times of flight.

TWO DIMENSIONAL CONTINUOUS CURVILINEAR SOLUTION

A review of the equations of motion pertaining to fixed wing and rotary wing aircraft flight mechanics and autopilot control of such aircraft has led to the conclusion that there will be significant improvements for certain tactical maneuvering scenarios, if targets are engaged by fire control systems having some form of offset predictor that accounts for total acceleration changes during the flight time of the projectile. Studies conducted by Air Force organizations building flight simulators and by analysts of gun fire control systems suggest that a likely candidate for such a prediction algorithm is a circular arc predictor. Total motion of the maneuvering fixed and rotary wing aircraft during a projectile time of flight must be considered compared to ground vehicles, where only the apparent motion perpendicular to the LOS is utilized.

This section extends the curvilinear solution concept to two dimensions. A three dimensional solution is a logical follow-on development. It is informative to look at the two dimensional planar problem to understand the essential difference between a circular and constant acceleration predictor. Assume that a maneuvering target is in a steady state maneuvering flight condition such that the resultant forces acting on the maneuvering vehicle are perpendicular to the vehicle's velocity in the plane of symmetry. A flight condition then exists, which is referred to as a coordinated maneuver. The path made by the vehicle when such a balance of forces exists, is a circle and the plane the circle lies in is referred to as the maneuver plane. The velocity is tangent to the maneuver plane circle, the acceleration is normal to the velocity and directed toward the center of the circle.

When this circular path is observed by a fire control system, the situation is as described in Figure 2.

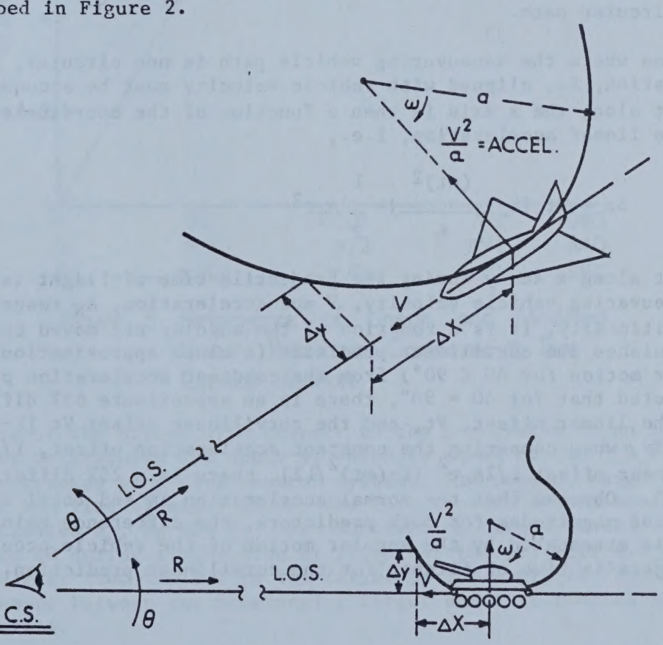


Figure 2. Maneuvering Target Scenarios.

If the circle radius is a, vehicle velocity V and acceleration A_N, then the angular velocity ω of the vehicle is V/a or A_N/V. The arc $\Delta\theta$ is a function of ωt where t is the projectile time-of-flight. If an x-y axis system is located with its origin centered at the target with x along the velocity V and the y axis along the acceleration A_N, movement during the projectile time of flight can be determined, with respect to the target centered x-y axis system.

The movement along the y axis is the same as in the one dimensional solution.

Movement along the x axis must also be considered in the two dimensional solution.

$$\Delta x = a \sin \omega t$$

or

$$\Delta x = a\omega t \, [1 - \frac{(\omega t)^2}{3!} + .]..$$

But $a\omega$ is the magnitude of the velocity v.

$$\text{Then } \Delta x = Vt[1 - \frac{(\omega t)^2}{6}].$$

Figure 3 shows the performance of both a linear and modified linear predictor for a circular path.

For cases where the maneuvering vehicle path is non circular, an additional acceleration, A_T, aligned with vehicle velocity must be accounted for. Movement along the x axis is then a function of the coordinated acceleration and this linear acceleration, i.e.,

$$\Delta x = Vt \, [1 - \frac{(\omega t)^2}{6}] + \frac{1}{2} A_T t^2$$

Movement along x and y during the projectile time-of-flight is a function of the maneuvering vehicle velocity, V and acceleration, A_N respectively, but additionally, it is a function of the angular arc moved through. This distinguishes the curvilinear predictor (a close approximation to the true circular motion for $\Delta\Theta < 90°$) from the constant acceleration predictor. It is noted that for $\Delta\Theta = 90°$, there is an approximate 65% difference between the linear offset, Vt, and the curvilinear offset $Vt\,[1 - (\omega t)^2/6]$. Similarly, when comparing the constant acceleration offset, $1/2A_N t^2$, and the curvilinear offset $1/2A_N t^2\,[1-(\omega t)^2/12]$, there is a 25% difference for a $\Delta\theta = 90°$. Observe that the normal acceleration A_N and total velocity V have fixed magnitudes for both predictors, the difference being that their effect is attenuated by the angular motion of the vehicle occurring during the projectile time of flight, for the curvilinear prediction.

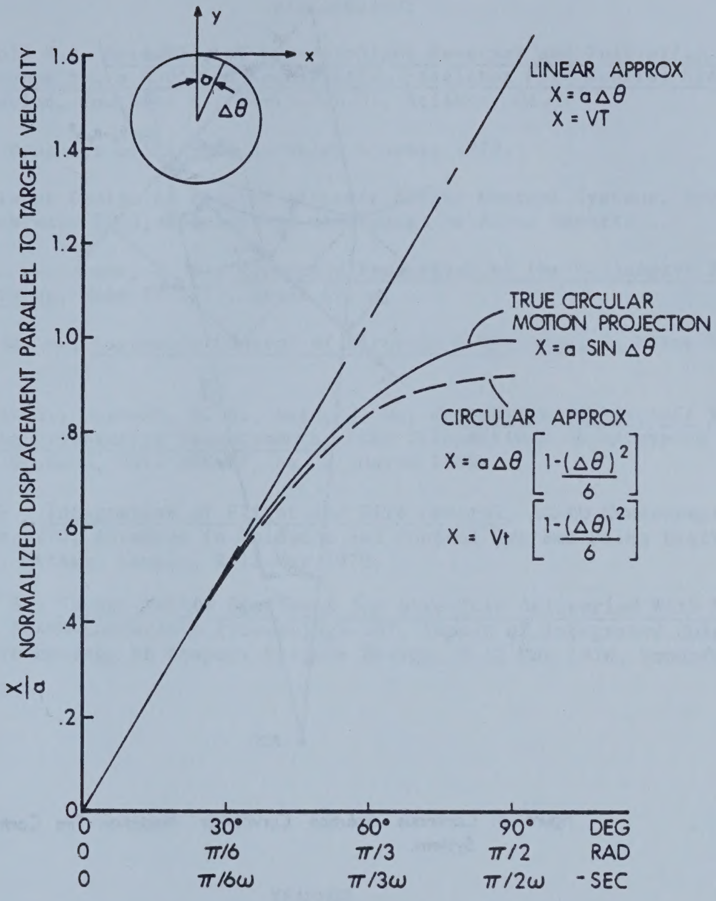


Figure 3. Performance of Linear and Modified Linear Predictors Compared to True Circular Motion.

Transformation of the offset with respect to the x-y system to an LOS reference is possible by determining the angle ϕ, which is equal to $\tan^{-1} R_p \dot{\sigma} / \dot{R}_p$. The target is tracked by a fire control system. The present range is $\dot{R}_p$, the motion of the line of sight is σ. The fire control system measures R_p, $\dot{\sigma}$ and estimates R_p, $\dot{R}_p$ and $\dot{\sigma}$. The maneuvering target moves through an arc ΔΘ during t, the time of flight of the projectile. The gun line is offset from the LOS to the target by the angle λ. The geometry of the engagement between the maneuvering target and fire control system is shown in Figure 4.

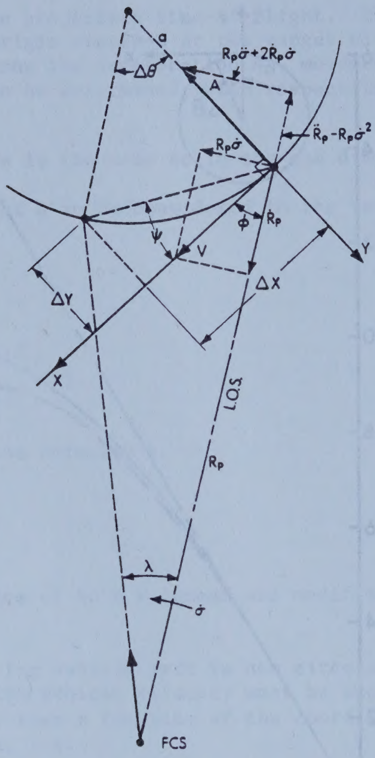

a
$\Delta\theta$
$R_p\ddot{\sigma}+2\dot{R}_p\dot{\sigma}$
A
$\ddot{R}_p-R_p\dot{\sigma}^2$
$R_p\dot{\sigma}$
ψ
ϕ
$\dot{R}_p$
V
Y
ΔX
ΔY
X
L.O.S.
R_p
λ
$\dot{\sigma}$
FCS

Figure 4. Continous Solution Curvilinear Prediction Fire Control System.

SUMMARY

Maneuvering land, sea and air vehicles exhibit velocities and accelerations that are closely aligned, respectively, with the longitudinal and lateral axes of the vehicle, especially when operating in regions of intended deployment. Gun fire control system performance may be improved when these types of maneuvering targets are engaged if the prediction process takes advantage of the fact that the predominate acceleration is created by rotation of the vehicle and oriented orthogonal to the longitudinal axis of the vehicle. Motion during projectile time-of-flight relative to a vehicle body axes system, fixed at time of fire, is better described by a circular arc predictor than a second order predictor for such maneuvering paths. For vehicle maneuvering levels and projectile times of flight which result in large errors along the velocity vector and relatively small errors perpendicular to the velocity of the vehicle, it may be advantageous to modify only the Vt offset.

ACKNOWLEDGEMENT

The author wishes to thank Mrs. Jo Ann Marderness for typing the manuscript.

BIBLIOGRAPHY

Burke, Harold H., Detection of Line-of-Sight Reversal and Initiation of Firing Commands for a Modified Acceleration Predictor Fire Control System, IEEE Convention, SouthCon 81, 13-15 Jan 81, Atlanta, GA.

Etkin, B., Dynamics of Flight, J. Wiley & Sons, 1979.

Fundamentals of Design of Piloted Aircraft Flight Control Systems, Vol. II; AE 61-4, February 1953, Dynamics of Airframes, Bu Aero, Report.

Chen, R. T. N., Jeske, J. A., Kinematic Properties of the Helicopter in Coordinated Turns, NASA TP 1773, April 81.

Blakelock, J. H., Automatic Control of Aircraft & Missiles, J. Wiley & Sons, 1965.

Kendrick, J. D., Maybeck, P. S., Reid, J. G., Estimation of Aircraft Target Motion Using Orientation Measurements, IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-17, No. 2, March 1981.

Huber, R. R., Integration of Flight and Fire Control, AGARD Conference Proceedings No. 272, Advances in Guidance and Control Systems Using Digital Techniques, Ottawa, Canada, 8-11 May 1979.

Ausman, J. S., Target Marker Placement for Dive-Toss Deliveries With Wings Non-Level, AGARD-Conference Proceedings-257, Impact of Integrated Guidance & Control Technology on Weapons Systems Design, 9-12 May 1978, Sandefjord, Norway.

A DESIGN METHODOLOGY FOR ESTIMATORS AND PREDICTORS IN FIRE CONTROL SYSTEMS

JAMES F. LEATHRUM
Department of Electrical and Computer Engineering
Clemson University
Clemson, South Carolina 29631

ABSTRACT

In the interest of establishing a systems approach to the design of fire control systems, the design of estimators and predictors is formalized into a direct procedure. The usual, trial-and-error approach to fixing the parameters of estimators is circumvented. The performance requirements of the fire control system are used at the outset as inputs to the computation of filter parameters. Component specifications are an important outcome of this design process. In particular, tracking accuracy requirements emerge as results of the design process. Tradeoffs between first and second order predictors are quantified and used to select the best predictor.

The state-of-the-art in the design of Kalman filters for fire control systems leaves the designer with several parameters to be used to overcome the effects of modeling errors. These parameters are the model and observer noise variances, and they are usually fixed by searching for satisfactory operating conditions. The methodology developed here uses miss distance and target bandwidth characteristics to fix the noise variances. The miss distance is a system performance requirement, and the bandwidths are obtainable from the analysis of broad classes of targets. The result is a direct design methodology which is free of searching.

THE ROLE OF ESTIMATORS IN FIRE CONTROL SYSTEMS

One of the fundamental processes which arises in gun fire control is the process of estimating the state of the target. This estimation process is readily discernible in even the least sophisticated systems. As the system design is augmented to include capabilities against maneuvering targets, the burden upon the estimation process becomes progressively greater both in terms of accuracy and number of states to be estimated. For instance, for straight-line, constant velocity targets, there is no need to estimate acceleration. On the other hand, the utility of a velocity estimate will depend directly upon the accuracy of the estimate. An inaccurate lead may be worse than no lead at all. The same argument holds for the higher derivatives of motion.

At any point in the evolution of fire control technology there is probably a practical limit to the dimensionality and accuracy of the estimation process. Considerations of processor speed, observer accuracy, and target identification (modeling) would be expected to determine the number and accuracy of the state variable estimates. If the estimator technology is critical in the sense that no other technology would inhibit the system implementation, the design strategy is one of achieving the most sophisticated system which can be supported by the estimator. In the current state-of-the-art, the estimator and observer (sight or tracking) technologies seem to share critical roles. The solution of the tracking part of the design problem will require much of the same technological resources as the estimator (i.e., processor speed and models). In addition, most of the additional cost of a highly sophisticated system will probably fall in the sight or tracking mechanism.

Acknowledging this set of interacting technologies, the purpose of this paper is to formalize some of the issues in terms of simple models of observers, estimators and predictors. The performance of predictors working in tandem with optimal estimators will be used to specify the parameters of the estimators and observers. Tradeoffs between observer technology and overall (predictor) performance will become apparent. The propagation of errors through typical predictors will be used to assess the trade-offs between Nth and N+1th order predictors.

A by-product of this work is the formulation of a direct design methodology for Kalman filters in fire control systems. The unknown noise statistics will be assessed in terms of the power spectra of generic targets and allowable performance limits of the predictor.

CONVENTIONAL DESIGN METHODOLOGY

The conventional approach to the design of estimators and predictors for fire control systems is best illustrated by the following development of models and parameters. One would start by formulating target and observer models of the form.

(a) Target Model

$$X_{k+1} = \Phi_k X_k + B_k U_k$$

(b) Observer Model

$$Y_k = H_k X_k + V_k$$

These models immediately involve a linearization approximation. The target model captures the well defined motion in the state transition matrix, Φ_k, and leaves the less defined part of the motion to a noise term, B_k, U_k. The observer is usually a statement that not all the state components are visible, and that the observations are corrupted by an error, V_k. (The index, k, is a discrete time index). If one can further approximate U_k and V_k by white gaussian, zero mean processes, an estimator of X_k can be formulated as:

$$\tilde{X}_{k+1} = \Phi_k \hat{X}_k \quad \text{(Predicted State)}$$

$$\hat{X}_k = \tilde{X}_k + K_k (Y_k - H_k \tilde{X}_k) \quad \text{(Corrected State)}$$

which is the Kalman Filter wherein

$$K_k = \tilde{P}_k H_k' (R_k + H_k \tilde{P}_k H_k')^{-1} \quad \text{(Filter Gain)}$$

$$\tilde{P}_{k+1} = \Phi_k \hat{P}_k \Phi_k' + B_k Q_k B_k' \quad \text{(Predicted Variance)}$$

$$\hat{P}_k = \tilde{P}_k - K_k H_k \tilde{P}_k \quad \text{(Corrected Variance)}$$

$$R_k = E(U_k U_k') \quad \text{(Observer Noise Variance)}$$

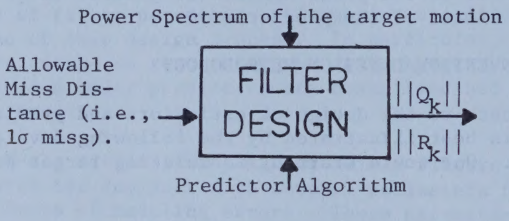
$$Q_k = E(V_k V_k') \quad \text{Model Noise Variance)}$$

In the most sophisticated fire control systems, the target noise is represented in a target oriented coordinate system. Thus, the given Q_k will rotate as the target moves which in turn leads to a nonsteady K_k. The Kalman gains tend to change throughout the estimation process. In addition, R_k may be range dependent which leads to further variability in K_k.

In designing such a filter, the implementor is left with choices of the magnitude of Q_k and R_k (i.e., $||Q_k||$ and $||R_k||$). A conventional design process would require assessing $||R_k||$ from the accuracy of the instrumentation used by the observer. Since $||Q_k||$ represents unmodeled behavior, it is usually adjusted to achieve some other objective, such as white innovation, or minimum ensemble miss distances. Whatever the objective, the last phase is unguided by the theory and thus usually requires extensive simulation to determine $||Q_k||$.

A DIRECT DESIGN METHODOLOGY

A methodology which could utilize maximum allowable miss distances to assess the design parameters directly would have some obvious advantages over the conventional process. Such a methodology is proposed here with the design features shown, i.e.,


Power Spectrum of the target motion
Allowable Miss Distance (i.e., 1σ miss).
FILTER DESIGN
||Q_k||
||R_k||
Predictor Algorithm

THE DIRECT DESIGN OF ESTIMATORS

The principles of the design will be illustrated by restricting the discussion to a single dimension and further restricting the models to

Φ : Upper Triangular

Q_k: Scalar constant, q

R_k: Scalar constant, r

These restrictions do not limit variability of the gains in the final design, but only allow one to focus attention to the magnitudes of the parameters in each direction. The design process requires solution of the steady state filter equations which become

$$\tilde{P} = \Phi\tilde{P}\Phi' + BqB' - \Phi\tilde{P}H' \ (r + H\tilde{P}H')^{-1}H\tilde{P}\Phi'$$

The solution for $\tilde{P}$ in terms of r and q requires iteration. However, a closed form solution for $\tilde{P}/q$ and r/q in terms of bandwidth of the target motion is possible. For instance, for a third order model, it requires the observation from the analogous continuous models that

$$P_{33}/q = (1/\omega_V) \cdot \Delta t$$

where ω_V is the bandwidth for velocity, and $\Delta t = t_{k+1}$ is the time increment between observations. This in turn, fixes $\tilde{P}_{33}/q$ by passing $\hat{P}_{33}/q$ through the predicted variance equation, i.e.,

$$\tilde{P}_{33}/q = (\Phi_{33})^2 \ \hat{P}_{33}/q + B_3{}^2$$

From this starting point, all the other variance ratios can be found. The required bandwidths can be assessed from the power spectrum observed in field tests of generic targets. The ratios ($\tilde{P}/q$ and r/q) completely specify the one dimensional, steady design, but they do not produce the magnitudes, $||Q||$ and $||R||$, needed in a multidimensional design. The required magnitudes are obtained from the variance of miss distance.

$$\frac{\sigma_{miss}}{\sqrt{q}} = (T \ \hat{P}/q \ T' \ )^{1/2}$$

In an optimal design, the $\hat{P}$ is interpreted as the variance of the estimator error. $\hat{P}/q$ is obtained from $\tilde{P}/q$ by passing the latter through the corrected variance equation. The vector is the coefficient vector for a predictor. Thus, the above equation represents the variance propagation through an Nth order predictor. Since $\hat{P}/q$ is available (from $\tilde{P}/q$), one can directly determine $\sigma_{miss}/\sqrt{q}$. A specification of σ_{miss} will lead to q thence to r (from r/q).

The logical outcome of this process is specification of q and r in terms of the filter bandwidth, the miss distance, and the predictor coefficients. Once r is found, the question is raised as to whether an observer with an accuracy of r is achievable. Thus, this design process may provide direct motivation for enhancing the observer technology.

PATHOLOGICAL DESIGNS

The design methodology outlined in the previous section must be approached with an appreciation that the results may not always be reasonable and achievable. The resulting $||R||$ and $||Q||$ may be thought of as intensities of noise on the observer and the target respectively. The $||R||$ may be so small that it will not be achievable using the evolving technology of sight mechanisms. Conversely, the $||Q||$ may be so large that targets cannot possibly achieve such an intensity of maneuver (i.e., rate of change of acceleration). Both situations represent pathological cases. In the best of situations, the $||Q||$ may be brought in bounds by reducing the variance of miss distance without seriously deflating $||R||$, thus not forcing new observer technology. Alternately, if the $||Q||$ is within bounds, the miss distance requirements may be relaxed to bring $||R||$ within current technological constraints.

In the worse case, both $||R||$ and $||Q||$ may be beyond reasonable bounds. Changing the miss distance requirements would not alleviate this pathology. The only recourse at this point is to reassess the frequency response of the filter. As the filter bandwidth is reduced, the ratio, $||Q||/||R||$, is reduced. Thus, in order to achieve reasonable noise intensities, it may be necessary to filter out some of the higher frequency components of the target motion. Such a redesign is in the direction of model mismatching and thus toward progressively less optimal estimation. A deterioration in actual miss distance (as opposed to propagated variances) is inevitable.

DESIGN OF PREDICTION ALGORITHMS

The prediction design problem is viewed as a trade-off between the errors incurred in propagating incorrect estimates through a predictor versus the errors incurred in failure to utilize an available estimate of a target state variable. The problem is similar to one which arises in numerical analysis wherein some situations "cancellation errors" may be larger than a computed result.

The design of predictors will be considered in the situation where the predictor exists in tandem with a Kalman estimator. This introduces no loss of generality, and yet, it is an acknowledgement of the current state-of-the-art. This structure does permit a presumption that state estimates and their variances are available for this analysis. The Kalman estimator would produce a sequence of estimates, $\hat{X}_k$, and variances of the estimation errors, P_k. If, at some time, it is necessary to predict ahead N steps without observations, the best, linear, unbiased estimator is

$$\hat{X}_{k+N} = \overset{N-1}{\underset{i=0}{\pi}} \phi_{k+i} \quad \hat{X}_k$$

which would be approximated by

$$\hat{X}_{k+N} = (\phi_k)^N \hat{X}_k$$

The variance of the estimation error at k+N is obtained from the recussion

$$P_{k+1} = \phi_k P_k \phi_k' + B_k Q_k B_k'$$

where $Q_k \equiv E(U_L U_k')$. Extrapolating this to k+N gives

$$P_{k+N} = \phi_k^N P_k (\phi_k^N)' + \sum_{j=0}^{N-1} \phi_k^j B_{k+j} Q_{k+j} B'_{k+j} \phi_k^{j'}$$

which will be viewed as composed of

$$P_{k+N} = \left\{ \begin{array}{l} \text{Propagated variance} \\ \text{of the estimates} \end{array} \right\} + \left\{ \begin{array}{l} \text{Error variance induced by} \\ \text{subsequent target motion} \end{array} \right\}$$

So far, this discussion seems to indicate that the design and analysis of the predictor of $\hat{X}_{k+N}$ is a closed issue and that the variances are left as a take-it-or-leave-it result. However, there is a choice to be made if one recognizes that the model is only one of a number of possible models. Furthermore, even if the models are correct, there may be a better, biased predictor. To formalize this notion, a predictor is defined as

$$\hat{X}_{k+N} = \phi^*_k X_k$$

where ϕ^*_k is not necessarily ϕ_k^N. The loss function of interest is the norm of the predictor error

$$||E_p|| = ||\hat{X}_{k+N} - X_{k+N}||$$

and the risk (which is to be minimized) will be

$$R(\phi^*) = E[||E_p||]$$

that is, we will seek that predictor algorithm, ϕ^*, which minimizes the expected norm of predictor error.

Although the problem just formulated is of interest in itself, there is an equally interesting dual. That is, given a maximum allowable risk, what predictor algorithm will lead to the best overall estimator-predictor performance? Here, performance is viewed in terms of constraints placed upon sight technology and target agility. The least constraining design will be viewed as the best design.

FIRE CONTROL SYSTEMS AS A SPECIAL CASE

The problem definition stated in the previous section leaves a large number of parameters to be considered in the general case. To illustrate the utility of the approach, the subsequent discussion will be limited to situations which arise in the design of fire control systems. The following limitations will be presumed:

a. Only the positional component of the predictor error will be included in the loss function.

b. The model to be employed in the estimator is of the form

$$\phi_k = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 & \cdots \\ & 1 & \Delta t & \cdots \\ \underline{0} & & 1 & \cdot \\ & & & \cdot \\ & & & & \cdot \end{bmatrix} \quad ; \quad B_k = \begin{bmatrix} \cdot \\ \cdot \\ \cdot \\ \cdot \\ \Delta t^3/6 \\ \Delta t^2/2 \\ \Delta t \end{bmatrix}$$

$$H_k = [\,1\ 0\ 0\ .\ .\ .\ \qquad 0\,]$$

The prediction algorithm will be of the form

$$[\phi^*]_{\text{Position}} = [\,1 \quad \gamma_V t_f \quad \gamma_A t_f^{\,2}/2\ .\ .\ .\ .\,]$$

where $t_f = N\Delta t$, and the γ_V, γ_A, etc., are the optimization parameters. The case of greater interest in current applications is the one with a 3-dimensional state vector

$$\phi_k = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \quad ; \quad B = \begin{bmatrix} \Delta t^3/6 \\ \Delta t^2/2 \\ \Delta t \end{bmatrix}$$

$$H = [\,1 \quad 0 \quad 0\,]$$

Here, the state vector is the position, velocity, and acceleration along a single axis of the target

$$X = \begin{bmatrix} x \\ v_x \\ a_x \end{bmatrix}$$

The prediction problem along each axis of the target can be treated independently for design purposes. In real-time, however, one would expect inherent coupling of the state equations along the respective axes. In other words, we will be generating up to three designs to be coupled together in a real-time implementation.

Concentrating upon a single coordinate axis, the loss function may be defined as

$$\varepsilon_p^{\,2} = (\hat{x} - x)^2_{k+N}$$

Here $||\varepsilon||^2$ is used as a loss function for conciseness. In turn the risk is

$$R(\phi^*) = E(\varepsilon_p)^2$$

Using the definitions of $\hat{x}$ and x as well as the notation

$$T = [1\ t_f\ t_f^{\,2}/2] = [\phi^N]_{\text{position}}$$

$$T^* = [1\ \gamma_V t_V\ \gamma_A t_f^{\,2}/2] = [\phi^*]_{\text{position}}$$

the risk condenses to

$$\begin{aligned} E(\varepsilon_p{}^2) = T^*P_kT^{*'} &+ T^*Z_k(T^*-T)' \\ &+ (T^*-T)Z'_kT^{*'} \\ &+ (T^*-T)E(X_kX_k{}')(T^*-T)' \\ &+ E\,(\varepsilon^2{}_{TI}) \end{aligned}$$

where $Z_k = E[(X_k-X_k)X'_k]$

and $E(\varepsilon^2{}_{TI})$ = Variance of the target induced error

Thus, the risk is composed of terms which are interpreted as

$E(\varepsilon_p{}^2)$ = Variance propagated from the estimation errors
+ Correlations between estimation errors and states (2 terms)
+ Mean squared magnitudes of the target states after propagation
+ Error variances induced by subsequent target motion

Since the last term is not a function of the optimization parameters, it will be treated as a constant. The optimization involves a trade-off between the other terms. Thus a new risk is formulated as the residual risk, $E(\varepsilon_R{}^2)$, due to the choice of the prediction algorithm.

$$E(\varepsilon_R{}^2) = E(\varepsilon_p{}^2) - E(\varepsilon^2{}_{TI})$$

In the first three terms of the risk equation, the filter parameters, P_k, Z_k, may be determined as ratios $P_k/_q$ and $Z_k/_q$ at steady state. These ratios, as well as r/q, are functions of a single bandwidth parameter, ω_V, for the target velocity. (Note that $q \equiv E(U^2)$; $r \equiv E(V^2)$). After dividing the risk equation by q, and gathering terms, the dual optimization problem becomes one of maximizing.

$$r/E(\varepsilon_R{}^2) = \frac{K(\omega_V).N(S,t_f,\gamma_V,\gamma_A)}{D(\omega_V,t_f,\gamma_V,\gamma_A)}$$

$$S = E(XX')/E(\varepsilon_R{}^2)$$

$$\begin{aligned} N(S,t_f,\gamma_V,\gamma_A) = 1-[S_{33}(t^2{}_f/_2)^2\,y^2 &+ S_{22}(t_f)^2.x^2 \\ &+ 2S_{23}{}'(t_f{}^3/_2).xy] \end{aligned}$$

$$\begin{aligned} D(\omega_V,t_f,\gamma_V,\gamma_A) = T\,(P/q)T' &- [(P_{33}/_q).(t_f{}^2/_2)^2y^2 \\ &+ (P_{22}/_q)(t_f)^2.x^2 + 2(P_{23}/_q)(t_f{}^3/_2).xy] \end{aligned}$$

where $x = 1-\gamma_V$ and $y = 1-\gamma_A$

$$K(\omega_V) = r/_q$$

At the steady state design condition, the cross correlations, $Z/_q$, are precisely the negative of the variance ratios, i.e.,

$$Z/_q = -P/_q$$

Given the target characteristics, specified in rough terms by S and ω_V, and the maximum allowable risk, the designer must choose γ_V and γ_A to maximize r. Maximizing r, in turn, maximizes q since q/r is a constant. Thus, the estimator-predictor combination is least contraining in respect to the sight technology and target agility.

The final form of the performance criterion is a rational function in x and y. Further direct analysis of the extrema of this function is possible, but some preliminary insights may be possible by taking cross-sections of the contours with respect to coordinate planes. This corresponds to examining single variable perturbations about the standard first and second order predictors.

SOME SPECIAL PREDICTOR DESIGNS

Two interesting special cases offer some insight into the dual optimization of the predictor design. In each, one of the design parameters will be set to its natural value (i.e., $\gamma=1$).

$$r/E(\varepsilon_R^2)\Big|_{y=0} = \frac{K(\omega_v)[1-S_{22}(t_f)^2x^2]}{T(P/q)T' - (P_{22}/q)(t_f)^2x^2}$$

$$r/E(\varepsilon_R^2)\Big|_{x=0} = \frac{K(\omega_v)[1-S_{33}(t_f^2/2)^2y^2]}{T(P/q)T' - (P_{33}/q)(t_f^2/2)^2y^2}$$

Both of those functions are of the same general form, and both are symmetrical about the origin of the free variable (i.e., symmetrical about $\gamma=1$). The general form is

$$r/E(\varepsilon_R^2) = C(\omega_v)\,\frac{[1-a(t_f,S)x^2]}{[1-b(t_f,\omega_v)x^2]}$$

where

$$C(\omega_v) = \frac{r/q}{T(P/q)T'}$$

$$a(t_f,S) = S_{ii}[t_f^{\,i-1}/(i-1)!]^2$$

$$b(t_f,\omega_v) = [(P_{ii}/q)/(T(P/q)T')].[t_f^{\,i-1}/(i-1)!]^2$$

This function is a ratio of parabolas whose shape is determined by the relative values of $a(t_f,S)$ and $b(t_f,\omega_v)$. A typical shape is shown in Figure 1. Here, the offset, x, represents a typical velocity correction.

UNREALIZABLE DESIGNS

It is evident from the general analysis of the optimal predictor designs, that some of the parameter values lead to negative values of $r/E(\varepsilon_R^2)$. One could argue that these are never cases of interest since they do not occur in the neighborhood of the natural parameters when $a<b$, and when $a>b$, the natural value is clearly optimal. Nonetheless, an interpretation of the results of when $r/E(\varepsilon_R^2)$ is negative is called for since the parameter values are realizable.

Since the desired risk, $E(\varepsilon_R^2)$, is asserted to a priori, the case of negative $r/E(\varepsilon_R^2)$ corresponds to situations where that risk is not achievable by any r, q combination. This may arise because either the magnitudes of the states are too large, the risk is too small, the time of flight is too large, or the variance of the estimation errors is too large. Of all of these conditions, the time of flight is the most pervasive since it effects both the pole and zero placement. Thus, the whole structure of the optimal design as determined by the relative magnitude of $a(t_s,S)$ and $b(t_s,\omega_V)$ may be significantly altered by the time of flight (i.e., the distance to the target).

CONCLUSIONS AND RECOMMENDATIONS

The results of the previous sections should be viewed as indicative of the potential screening power of a design methodology based upon propagated errors. Critical choices in predictor parameters are possible and performance characteristics are discernible.

Given the pervasive influence of the time of flight, t_f, on the results obtained herein, one must conjecture at this point about the real time features of the methodology. One could feed forward smoothed estimates V and A and compute the optimal γ_V and γ_A during an engagement. The potential gain from such a scheme is so great that it deserves some testing using a fire control simulator.

It is apparent that the major predictor design issues may vary greatly with the intensity of the target motion. For larger leads, the importance of the correct modeled nominal motion cannot be overemphasized. From one model to the next there may be significant differences in sensitivity and propagated error as well. In any case, for large leads, there is seldom an opportunity to use other than the natural values of γ_V and γ_A.

On the other hand, for less intense motion, situations may arise where propagated error completely dominates a correction. Whole terms may drop out of the predictor as a result. Here, however, the differences between nominal models is not as significant.

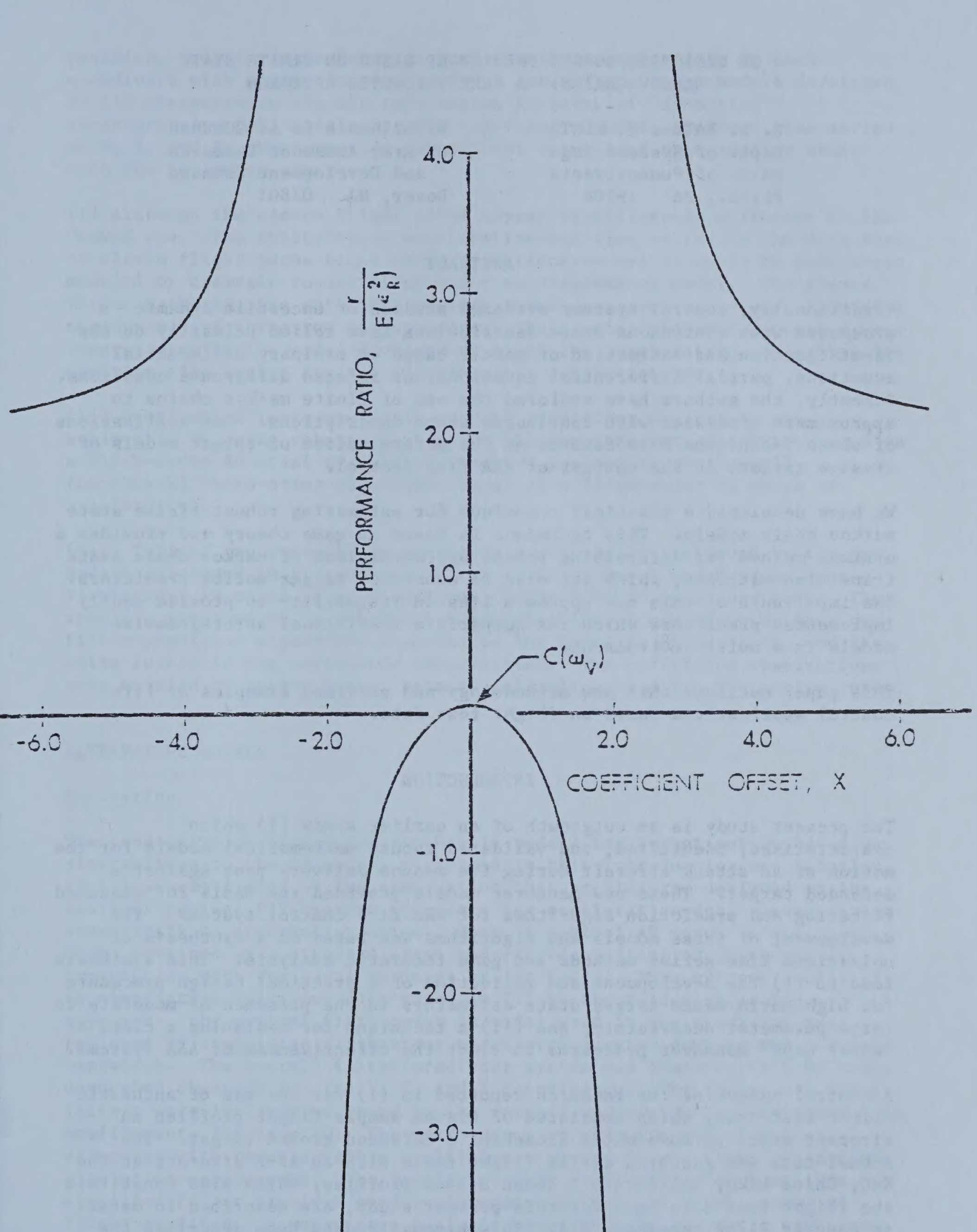


Figure 1: A Typical Contour of a Predictor Performance Function

ON DESIGNING ROBUST PREDICTORS BASED ON FINITE STATE MARKOV CHAINS: A GAME THEORETIC APPROACH

E. B. PATE & M. MINTZ
Dept. of Systems Eng.
Univ. of Pennsylvania
Phila., PA 19104

W. DZIWAK & S. A. GOODMAN
US Army Armament Research
and Development Command
Dover, NJ 07801

ABSTRACT

Traditionally, control systems oriented studies of uncertain dynamic processes with continuous state descriptions have relied primarily on the identification and estimation of models based on ordinary differential equations, partial differential equations, or related difference equations. Recently, the authors have explored the use of finite markov chains to approximate processes with continuous state descriptions. Our applications of these techniques have focused on the determination of robust models of evasive targets in the context of AAA fire control.

We have developed a practical technique for estimating robust finite state markov chain models. This technique is based on game theory and provides a minimax method for determining robust approximations of markov chain state transition matrices, which are used to construct target motion predictors. The importance of this new approach lies in its ability to provide easily implemented predictors which can outperform traditional autoregressive models in a noisy environment.

This paper outlines this new methodology and provides examples of fire control applications based on flight test data.

INTRODUCTION

The present study is an outgrowth of an earlier study (1) which characterized, identified, and validated robust mathematical models for the motion of an attack aircraft during its weapon delivery pass against a defended target. These new maneuver models provided the basis for enhanced filtering and prediction algorithms for AAA fire control systems. The development of these models and algorithms was based on a synthesis of univariate time series methods and game theoretic analysis. This synthesis lead to (i) the development and validation of a practical design procedure for high performance target state estimators in the presence of moderate to large parameter uncertainty, and (ii) a technique for designing a class of "worst case" maneuver processes to blunt the effectiveness of AAA systems.

A central aspect of the research reported in (1) was the use of authentic flight test data, which consisted of eleven sample flight profiles an aircraft might perform while attacking a defended ground target. The actual data was gathered during flight tests with an A7-E aircraft at the NWC, China Lake, California. These attack profiles, which also constitute the flight test data base for this present study, are described in detail in Chapter II of reference (1). This kinematic data base describes the aircraft motion in a cartesian coordinate system, where the origin of this coordinate system is the aircraft's intended target, as well as the assumed location of the AAA weapon system. The kinematic data describing the eleven flight profiles in the XYZ coordinate system includes consistent position, velocity, acceleration, and acceleration-dot data in each coordinate with a time increment of 0.1 sec. The primary models developed in (1) characterize the aircraft motion in terms of "decoupled" autoregressive (AR) models for the individual acceleration-dot time series in X, Y, and Z. We summarize the salient results of this earlier study with the following remarks:

(i) Although the eleven flight paths appear significantly different to the "naked eye," the thirty-three acceleration-dot time series in the data base -- eleven flight paths times three directions -- are shown to be accurately modeled by a single robust fifth-order autoregressive model. The eleven flight paths in this data base include three dive toss maneuvers, five dive maneuvers, and three pop-up maneuvers. The acceleration-dot processes were incorporated in the model development since the acceleration, velocity, and position time series are all significantly nonstationary.

(ii) Substantial improvements in overall prediction capability are achievable by using robust, high-order filter-predictor algorithms based on a fifth-order AR model of acceleration-dot instead of the "usual" (benchmark) third-order algorithms based on a first-order AR model of acceleration.

(iii) Typical improvements in average hit probability achieved by the new models developed in (1) versus the standard benchmark model based on a first-order AR model of acceleration, ranged from 25 to 35 per cent. The specific enhancement in average hit probability associated with these new filter-predictor algorithms depended on the specific flight path, and the noise levels in the unfiltered observations. The unfiltered observations were modeled by target range, azimuth, elevation, and the respective rates.

ALTERNATIVE MODELS

Motivation

The consideration of dynamic stochastic models for target motion based on alternatives to the AR models described in this Introduction and detailed in (1) was motivated by the observation that predictors, designed on the basis of nominal fifth-order AR models for acceleration-dot, showed extreme sensitivity to observation noise. Here, a nominal AR model refers to a model obtained by a standard least squares identification procedure in conjunction with the usual goodness of fit tests. This extreme sensitivity to observation noise was eliminated in (1) by the following ad hoc approach: the nominal fifth-order AR model used in the predictor was replaced by a modified fifth-order model with greater damping and reduced bandwidth. The overall filter/predictor system was characterized by three decoupled channels for the X, Y, and Z coordinates. The individual Kalman filters were implemented based on the nominal values of the X triple-dot AR coefficients for Pass 10. The performance of this simplified filter structure with these specific coefficients was studied extensively through simulation experiments. The results of this study indicated that this simplifed filter provided good performance against all of the observed flight profiles. A worst case analysis based on game theoretic techniques revealed that this simplified nominal filter should be replaced by a structurally identical filter with different AR parameter values to counter plausable worst case maneuver processes which were not part of the flight test data base. We refer the reader to (1) for a complete description of this worst case analysis and filter design technique. The coefficients for the individual X, Y, and Z coordinate predictors were obtained by adjusting the overall bandwidth and damping to achieve good prediction performance based on the filtered observations. A common choice of modified AR coefficients was implemented for each of the three predictors. This adjustment of predictor bandwidth and damping necessitated some empirical analysis via simulation studies. It is worth emphasizing that these "decoupled" eighth-order nominal filter/modified predictors provided 25 - 35 per cent improvement in average hit probability compared with the usual "decoupled" third-order benchmark algorithm. We note that although the nominal predictors based on fifth-order AR models of acceleration-dot showed extreme sensitivity to observation noise, which necessitated modifying the predictor's frequency response, a nominal predictor based on the Pass 10 X triple-dot AR coefficients provided excellent prediction capability against each of the eleven flight paths in a noiseless environment. This substantial sensitivity to observation noise is of more than academic interest. It suggests that the underlying process (truth model) which actually characterizes the aircraft motion may exhibit structural details which are not embraced by AR or more generally by autoregressive integrated moving average (ARIMA) approximations. This last statement is not intended to imply that ARIMA models are unsuited for maneuver process modeling, but rather that, in the set of all possible maneuver process models, there may be alternative classes of models which provide better explainations of the underlying process. One recognizes, in applications oriented model building, that there is a delicate balance which must be sought between process explaination (complexity) and implementation approximations (simplicity). The choice of pivot point for this balance is strongly influenced by the model's intended application.

MARKOV CHAIN MODELS of A-DOT

In this section we describe the determination and validation of nominal finite state markov chain approximations to the individual X, Y, & Z acceleration-dot processes for the flight paths in our data base. Our motivation for considering this class of models/approximations was based initially on the recognition that the normal acceleration process for each flight path was well-modeled by a piecewise linear process whose segments exhibited random slope and duration. Our analysis of the normal acceleration process was part of a detailed examination of alternative models for evasive attack aircraft motion based on aerodynamic variables and aircraft aspect. The results of this comparative analysis, in addition to the aerodynamic and pilot oriented considerations which justify the applicability of these various alternative models, are contained in (2).

The following approach is justified if we assume that the underlying A-dot data is generated by a (possibly) infinite state wide-sense stationary markov process with sample paths that exhibit piecewise constant behavior over intervals of random duration. We refer the reader to reference (2) for a more rigorous probabilistic description of the class of underlying processes for which our approximation algorithm is theoretically valid. In the following discussion we focus our attention on a generic sample path, for example, the X triple-dot realization for Pass 1.

An Approximation Algorithm

Our approximation algorithm has the following steps:

Step 1 - Dynamic Range: Determine the dynamic range (maximum & minimum values) of the time series data.

Step 2 - State Assignment: Determine a finite set of "states" for the time series data.

Step 3 - Transition Matrix Estimation: Estimate the probabilities in the one-step state transition matrix -- based on the state assignments made in Step 2.

Step 4 - Repeat Steps 1 - 3 for the remaining time series data sets.

Step 5 - Hypothesis Testing: Perform the appropriate statistical tests to evaluate the validity of the hypothesis that the derived finite state markov chains have a common underlying distribution. (We refer the reader to reference (3) for an excellent treatment of hypothesis tests for finite state markov chains.)

Some Comments on the Algorithm

Step 2: There is no canonical method to resolve the state assignment question. We have explored two methods in this preliminary analysis. These methods are: (a) the method of uniform dynamic range, and (b) the method of uniform duration. The method of uniform dynamic range breaks the dynamic range of the time series into M equal intervals. (In this analysis we chose M = 5.) The center point of each interval was defined to be the value of the associated state. The method of uniform duration selects the M "state intervals" such that the number of "occupancies" in each interval are identical (or approximately identical). Here again, the value of M was chosen to equal 5, and the center point of each interval was used to designate the state value. The selection of M = 5, as well as the state assignment procedure represents judgmental calls on the part of the investigators. Further research on these issues is currently being pursued.

Step 3: The underlying one-step state transition matrices were estimated using the method of maximum likelihood (3).

Structural and Statistical Results

Although the flight paths all look very different to the "naked eye," the estimated one-step state transition matrices showed remarkable similarity between the X, Y, & Z directions for a given flight path, and between all of the flight paths. This result corresponds to the related behavior of the acceleration-dot spectra reported in (1). Use of the uniform dynamic range rule for state assignment led to one-step transition matrices which exhibit a pronounced birth-death behavior.

Prediction Using Markov Chains

Prediction based on a MMSE criterion is easily carried out for a finite state markov chain model. The propagation of the conditional expectation of the state n steps into the future is obtained by raising the one-step state transition matrix to the nth power and then multiplying the resulting matrix by a column vector of state values. The desired value of the conditional expectation is merely a weighted sum of n-step state transition matrix entries.

Numerical Results

In this section we present the results of a series of simulation experiments which were carried out to test the validity of this finite state markov chain approximation concept. The predictions are based on a "multiple integration" of the predicted values of acceleration-dot, acceleration, and velocity in each coordinate. A standard Taylor series model defines the overall predictor structure. In this simulation experiment the markov chain model was exercised against the benchmark predictor. Since sensitivity to observation noise is a critical issue, predictor performance with and without noise has been investigated. The signal to noise ratios used in these experiments correspond roughly to the low level noise figure experiments performed in (1). In the current experiments, uncorrelated observation noise was added to the data prior to the prediction calculations. No filtering was carried out on these data prior to the prediction calculations. Since Kalman filters were employed in the experiments reported in (1), a complete correspondence between these preliminary experiments and the results reported in (1), cannot be made. The results of the present experiments should be viewed as preliminary, since the markov chain predictors which were implemented in each case were based solely on the Pass 1 time series data. This choice represents a first cut, and illustrates the excellent performance of the markov chain approximation, which did not require any tuning. The results of the prediction experiments are contained in Table 1. We remark that the markov chain model performed well against the benchmark model, and exhibited significant noise immunity. The performance of the nominal (non robust) fifth-order AR models for acceleration-dot yield corresponding average hit probabilities significantly less than 0.1 at this noise figure.

The reader might wish to speculate as to what would result from a time series analysis of the five-state markov chain approximations to the original A-dot data. We performed this analysis and summarize the results as follows: (i) These time series are well-modeled as ARMA(2,2) processes and alternatively can be reasonably well represented by an AR(5) model. (ii) The incorporation of this AR(5) approximation in a standard predictor structure leads to a predictor which (a) outperforms the markov chain model in the noiseless case, (b) exhibits performance comparable to that of the markov chain model at low to medium noise figures, and (c) is outperformed by the markov chain approximation at higher noise figures. We remark parenthetically that the markov chain model tends to do better at the medium to high noise figures due to its quantized structure; whereas, at the zero or very low noise figure, the five-state quantization introduces small errors. We conclude these remarks by noting that the AR(5) approximation to the five-state markov chain approximation has a set of roots (pole pattern) which corresponds closely to the roots of the empirically determined "robust" AR(5) models obtained in (1).

Further results relating to the robustization of these preliminary markov chain approximations are reported in (2). Reference (2) includes a worst case analysis of the markov chain approximation process, as well as the complete details of the saddle-point theory which justifies the worst case analysis. An outline of the relevant saddle-point analysis is presented in the Appendix to the present paper.

CONCLUSIONS

This study has presented substantial evidence to support the assertion that finite state markov chain approximations can be used in place of AR (and ARIMA) models as the basis for predictors in a realistic AAA fire control application. The value of the markov chain approximations lie in their substantial immunity to observation noise. It has been shown previously in (1) that high order AR models can provide a substantial enhancement in prediction performance in comparison with the usual decoupled third-order benchmark model. However, the application of these high order AR models requires that the nominal parameter values be adjusted to reduce the bandwidth and increase the damping of the given predictor. Prior to the development of this finite state markov chain approximation technique, this adjustment procedure required a degree of numerical experimentation to obtain a suitable design. Whereas, the markov chain approximations need essentially no tuning to obtain a design with excellent observation noise immunity, and therefore these models can be used "directly off the shelf".

REFERENCES

(1) S. F. Huling, and M. Mintz, "ENHANCED FILTERING & PREDICTION FOR AAA FIRE CONTROL: An Application of Game Theory and Time Series Analysis," Final Report on Contract No. DAAK10-78-C-0186, Submitted to US Army ARRADCOM, May 1980. Published as Contractor Report ARSCD-CR-80015, AD-E400 507, November 1980.

(2) E. B. Pate, and M. Mintz, "NEW TECHNIQUES FOR DESIGNING ROBUST PREDICTORS FOR AAA FIRE CONTROL SYSTEMS: An Application of Markov Chain Models and Game Theory," Final Report on Contract No. DAAK10-79-C-0232.

(3) T. W. Anderson, and L. A. Goodman, "Statistical Inference About Markov Chains," Annals Of Mathematical Statistics, Vol. 28, 1957, pp. 89-110.

(4) D. Blackwell, and M. A. Girshick, Theory Of Games And Statistical Decisions, Wiley, New York, 1954.

APPENDIX

A Minimax Prediction Problem

Since the flight test data base was limited to eleven attack profiles, it is of interest to consider what techniques might be employed to design a predictor, based on the finite state markov chain concept, which would provide protection against a realistic "worst case" attack profile which is not represented in the actual data base. In the previous discussion, we described the identification of finite state markov chain approximations to the individual X, Y, and Z triple-dot data sets for a generic flight path. By applying this identification-approximation technique one obtains a set of thirty-three stochastic matrices which represent the individual one-step state transition matrices for the X, Y, and Z triple-dot data sets for the eleven flight paths.

A possible approach to this worst case analysis suggests seeking a least favorable choice of state transition matrix from a suitable set generated by and containing these thirty-three sample stochastic matrices. A reasonable choice of "uncertainty set" for this decision problem might be the convex hull of these thirty-three sample one-step state transition matrices. In what follows we formulate an abstract minimax prediction problem which captures the contextual essence of this AAA fire control scenario.

Notation

Let $S = \{ s_1 , s_2 , \ldots, s_r \}$ denote the state space of a finite state markov chain, where each $s_i \varepsilon R$ and $s_1 < s_2 \ldots < s_r$.

Let $V = \{ Q_\theta : \theta \varepsilon \Omega \}$ denote a family of rxr regular stochastic matrices.

Let q_θ denote the vector of steady state probabilities associated with the one-step state transition matrix Q_θ.

Let $s(k)$ denote the system state at time instant k, $k = 1, 2, \ldots, N$.

Let $\underline{s} = (s(1), s(2), \ldots, s(m))^T$ denote an observed sequence of state transitions over time instants $k = 1, 2, \ldots, m$.

Let $P_\theta (s(n), \underline{s})$ denote the joint probability distribution of $s(n)$ and $\underline{s}$, conditioned on Q_θ, where $m < n$.

Let $\delta (\underline{s})$ denote a predictor for $s(n)$ based on the observed sequence $\underline{s}$.

Let $r(\delta,Q_\theta)$ denote the conditional risk in predicting $s(n)$ based on $\underline{s}$ and conditioned on Q_θ, under quadratic loss.

Here

$$r(\delta,Q_\theta) = \sum \sum (s(n) - \delta(\underline{s}))^2 P_\theta(s(n), \underline{s}),$$

where the indicated summations are over all possible $s(n)$ and $\underline{s}$.

Preliminary Observations

(i) If V denotes the convex hull of a finite set of regular rxr stochastic matrices, then V is isomorphic to a compact convex subset of E^{r^2}.

(ii) Since we need only consider admissible decision rules δ, the range space of δ is naturally restricted to the closed interval $[s_1, s_r]$. Further, since there are exactly r^m observation sequences $\underline{s}$, the set of decision rules D can be identified with the r^m - fold cartesian product of $[s_1, s_r]$ with itself. Hence D is a compact convex subset of E^{r^m}.

(iii) The probability function $P_\theta(s(n), \underline{s})$ is a continuous function of Q_θ.

(iv) For fixed $Q_\theta \varepsilon V$, the conditional risk function $r(\delta,Q_\theta)$ is a continuous convex function of $\delta \varepsilon D$.

(v) For fixed $\delta \varepsilon D$, the conditional risk function $r(\delta,Q_\theta)$ is a continuous function of $Q_\theta \varepsilon V$.

A Saddle-Point Theorem

The abstract game defined by the triple $(r(\delta, Q_\theta), D, V)$ has a saddle-point (δ^*, μ^*), where δ^* is the desired minimax predictor, and μ^* is a mixed strategy on V with at most $(r^m + 1)$ points of support.

Proof: The proof of this theorem follows from Corollary 1 of Theorem 2.5.1 of reference (4) on the basis of Preliminary Observations (i) - (v).

AVERAGE HIT PROBABILITIES

CASE I - NOISELESS DATA

PASS NO.	PREDICTOR MODEL 5-STATE MARKOV CHAIN APPROX. FOR A-DOT PASS 1	BENCHMARK
1	.397	.322
2	.331	.226
3	.412	.309
7	.403	.352
10	.409	.345
14	.396	.299

CASE II - NOISY DATA

PASS NO.	PREDICTOR MODEL 5-STATE MARKOV CHAIN APPROX. FOR A-DOT PASS 1	BENCHMARK
1	.348	.317
2	.288	.214
3	.344	.282
7	.352	.324
10	.362	.327
14	.351	.288

Table 1

A NEW APPROACH TO SYSTEM DIAGNOSIS*

KAZUO NAKAJIMA
Department of Electrical Engineering and Computer Science
Texas Tech University
Lubbock, Texas 79409

ABSTRACT

We consider the problem of automatic fault diagnosis for self-testing systems. We propose a new approach, and show that it requires fewer tests and a simpler diagnosis algorithm than the approach introduced by Preparata, Metze and Chien.

1. INTRODUCTION

Consider a self-testing system S consisting of n units and a prescribed connection assignment of test links which assigns each unit to test a subset of the other units. The system S can be represented by a digraph $G(U,T) = G$, where $U = \{u_1,u_2,\ldots,u_n\}$ is the set of vertices of G correspondng to the n units and $T = \{(u_i,u_j)|u_i,u_j \varepsilon U$ and u_i ($\neq u_j$) is assigned to test $u_j\}$ is the set of tests which are represented by directed edges in G. The test outcomes are either "pass" (fault-free) or "fail" (faulty) and thus expressed as binary weights $w(u_i,u_j)$ of the edges (u_i,u_j) in T. In particular, $w(u_i,u_j) = 0$ if u_i judges u_j to be fault-free and $w(u_i,u_j) = 1$, otherwise. We assume that the test outcome $w(u_i,u_j)$ is reliable only if u_i is a fault-free unit.

The model described above, termed the PMC model, was originally proposed by Preparata, Metze and Chien [1]. Influenced by the theory of error-correcting codes, they defined the notion of t-diagnosable systems, designed for the purpose of diagnosing (identifying) faulty units; a system S is called t-diagnosable if given any test results, all faulty units in S can be identified, provided the number of faulty units present does not exceed t. We will refer to such a system as deterministically t-diagnosable, in short, d-t-diagnosable. Preparata et al. showed that if a system S of n units is d-t-diagnos-

---

*This work was supported by the Joint Services Electronics Program, Texas Tech University, under ONR Contract 76-C-1136.

able, then $n \geq 2t+1$ and each unit must be tested by at least t other units in S. Hakimi and Amin [2] then showed that these conditions are in fact sufficient if no two units test each other in S. They also gave a necessary and sufficient condition for a general system to be d-t-diagnosable.

It is clear that to construct a d-t-diagnosable system S, one needs at least nt tests. Several optimal constructions of d-t-diagnosable systems that require exactly nt tests have been proposed [1,3]. Assuming a uniform cost of c for each test, it would cost at least ntc to obtain the test results. The cost, ntc, may be excessive; furthermore, this is not the entire cost of the diagnosis process, as one must analyze the known test results to identify the faulty units in S. This leads to a definite need for simple diagnosis algorithms. Several diagnosis algorithms have been reported [3-8], but none are sufficiently simple. Among these, the algorithm proposed by Madden [6] is reported [3] to be the only one that can be applied to a general d-t-diagnosable system, but it has time complexity $O(nt^2)$ and requires a large amount of storage. The other algorithms are either partly incorrect [5] or only applicable to a subclass of d-t-diagnosable systems [3,4,7,8] and have time complexity, at best, $O(nt)$.

Motivated by the recent work in analog fault diagnosis by Wu, Nakajima, Wey and Saeks [9], we propose a new approach to the fault identification problem which requires fewer tests and a simpler diagnosis algorithm. Unlike deterministic system diagnosis, in our approach a sequence of tests is not predetermined but will be decided by using the test results previously obtained. In this sense it is called an adaptive system diagnosis. We define a system S to be adaptively t-diagnosable (in short, a-t-diagnosable) if there is a sequence of tests allowable in S which can identify all faulty units, provided the number of faulty units present does not exceed t. A d-t-diagnosable system is a-t-diagnosable but the converse is not always true.

In this paper, we present a method (called the a-t-d method) of determining a test sequence for an a-t-diagnosable system of n units in which, for simplicity, each unit is assumed to be able to test any other unit. This assumption requires $n(n-1)$ test links in the system and may not be practical in some cases. However, one actually needs $(t+1)(n+t-1)$ test links, which is comparable with the fact that one needs at least nt test links in a d-t-diagnosable system. In the a-t-d method one needs at most $t(t+1)$ tests to locate a fault-free unit and then at most n-1 additional tests performed by this fault-free unit. Therefore, the total number of tests is at most $n+t^2+t-1$, while in deterministic system diagnosis the least number of required tests is nt. Since t is usually much smaller than n, the number of tests in the a-t-d method is considerably less than the number of tests in deterministic system diagnosis. To locate a fault-free unit from the results of $t(t+1)$ tests, the a-t-d method requires $O(t^2)$ steps, which is much better than the best time complexity $O(nt)$, among the known diagnosis algorithms for d-t-diagnosable systems.

A a by-product the a-t-d method gives a construction for a d-t-diagnosable system which requires $(t+1)(n+t-1)$ tests (i.e., test links). This is not optimal in the sense that one needs more than nt tests. However, the diagnosis algorithm for this system is very simple and has time complexity $O(t^2)$, which is superior to the other algorithms.

In the next section, we state a fundamental theorem on a-t-diagnosable systems, which is an immediate consequence of Theorem 1 of Preparata, Metze and Chien [1]. A simple example is considered in Section 3, and a basic idea is described on how to locate a fault-free unit from $t(t+1)$ test results. Section 4 gives a formal description of the a-t-d method. It also provides an outline of the proof of its validity. In Section 5, we illustrate a construction for a d-t-diagnosable system which requires $(t+1)(n+t-1)$ test links. More discussion on adaptive system diagnosis is given with the help of examples in Section 6. The final section concludes the paper with further research directions.

2. ADAPTIVE SYSTEM DIAGNOSIS

As mentioned in the previous section, a d-t-diagnosable system is a-t-diagnosable. Therefore, we can immediately obtain the following fundamental theorem from Theorem 1 of Preparata, Metze and Chien [1].

Theorem 1: Let S be a system consisting of n units in which each unit is capable of testing any other unit. Then, S is a-t-diagnosable if and only if $n \geq 2t + 1$.

3. AN EXAMPLE

Consider a system S consisting of n = 10 units $u_1, u_2, \ldots, u_{10}$ in which each unit can test any other unit and there are no more than t = 3 faulty units. Since $n \geq 2t + 1$, S is a-t-diagnosable due to Theorem 1. We now want to find all faulty units in S. We first select a group G_1 of t = 3 units, say u_1, u_2, and u_3, and another group G_2 of t + 1 = 4 units, say u_4, u_5, u_6, and u_7. Then the units in G_1 test the units in G_2. Suppose we obtain the test outcomes in the form of a tx(t+1) matrix, D, as shown in Fig. 1, where the (u_i, u_j) element, $D(u_i, u_j)$, $(1 \leq i \leq 3,\ 4 \leq j \leq 7)$ indicates $w(u_i, u_j)$; that is, $D(u_i, u_j) = 0$ (resp. 1) if unit u_i evaluates unit u_j to be fault-free (resp., faulty).

G_2 G_1	u_4	u_5	u_6	u_7
u_1	0	1	0	0
u_2	0	0	1	1
u_3	0	1	0	1

Fig.1. Sample test outcomes.

We can conclude from the matrix of test outcomes that unit u_4 is fault-free. Suppose u_4 is faulty. Then, units u_1, u_2 and u_3 must be faulty, since they all judged u_4 to be fault-free. Therefore, we have t + 1 = 4 faulty units u_1, u_2, u_3 and u_4, which is a contradiction to the hypothesis that there are no more than t = 3 faulty units.

We can also conclude that u_6 is a fault-free unit. Suppose otherwise. Then units u_1 and u_3 must be faulty, and hence we already have t = 3 faulty units, u_1, u_3 and u_6. Therefore, unit u_2 must be fault-free, which implies that u_7 is a faulty unit, a contradiction.

Since u_4 and u_6 are fault-free units, one of them can test the remaining units and identify the faulty units in S. This terminates our process of adaptive system diagnosis.

In the next section, we show that it is always possible to locate a fault-free unit in an a-t-diagnosable system by analyzing t(t+1) test outcomes.

4. THE A-T-D METHOD AND ITS VALIDITY

As indicated in the previous section, the main frame of our procedure for diagnosing an a-t-diagnosable system is as follows:

(1) Select two groups G_1 of t units and G_2 of t+1 units.
(2) Perform t(t+1) tests of the units in G_2 by the units in G_1.
(3) Analyze the test outcomes and locate a fault-free unit.
(4) Perform n-1 tests by the fault-free unit found in (3).

We call the above procedure the a-t-d method. The number of tests performed in the above method is exactly $n+t^2+t-1$. As is seen in Sections 3 and 6, however, it is possible to find more than one fault-free unit in step (3), thus reducing the number of tests in step (4); or to locate a fault-free unit before finishing all of the t(t+1) tests in step (3). Thus, the number of tests required in the a-t-d method is usually less than $n+t^2+t-1$.

We now describe an algorithm for locating a fault-free unit from the t(t+1) test outcomes, and outline the proof of the correctness of our diagnosis algorithm.

Suppose that we are given t(t+1) test outcomes in the form of a tx(t+1) 0,1 matrix D. The t rows correspond to the units in G_1 and the t+1 columns to the units in G_2. The (i,j) element of the matrix, D(i,j), is 0 (resp., 1) if the i-th unit in G_1 finds the j-th unit in G_2 to be fault-free (resp., faulty). Let $R_0(i)$ and $R_1(i)$ (resp., $C_0(j)$ and $C_1(j)$) denote the total numbers of 0's and 1's in the i-th row (resp., j-th column) of the matrix. Then our algorithm is described as follows:

```
(1) Compute $R_1(i)$, $C_0(j)$ and $C_1(j)$ for i=1,...,t and j=1,...,t+1.
(2) for j = 1 to t + 1 do
        if $C_0(j) = t$ then return(j)
        Sort all $R_1(i)$ with D(i,j)=1 and obtain the sequence
        $i_{j1}, \ldots, i_{C_1(j)}$ such that $R_1(i_{jk}) \geq R_1(i_{j(k+1)})$ for k=1,
        ..., $C_1(j)-1$ (if $C_1(j)=1$, sorting is unnecessary and the
        sequence consists of $i_{j1}$ only).
        for k = 1 to $C_1(j)$
            if $C_0(j) + R_1(i_{jk}) \leq t - k + 1$ goto L
        end
```

```
    return(j)
L:  end
```

Using a bucket sort [10], the sorting can be done in $O(t)$ time for each j in Step (2). Then, it is easily seen that the time complexity of the algorithm is $O(t^2)$.

Lemma 1: The above algorithm always terminates with some column index returned.

Proof (sketch). When $C_0(j) = t$ for some j, the lemma is trivial. Suppose that $C_0(j) < t$ for $j=1,\ldots, t+1$. This implies that $C_1(j) > 0$ for $j=1,\ldots, t+1$. We note that the number of rows is smaller than that of columns in the matrix D. Using this fact, it can be shown that there is some column j such that for each $k=1,\ldots, C_1(j)$, $C_0(j)+R_1(i_{jk}) > t-k+1$. Therefore, the algorithm returns this column index j and terminates. []

Lemma 2: Let j be the column index returned by the algorithm. Then the unit corresponding to the j-th column is fault-free.

Proof. Suppose the unit is faulty. We first consider the case where $C_0(j) = t$. Then all of the t units corresponding to the rows are faulty, since they judge the unit to be fault-free. We have now t+1 faulty unit, which is a contradiction to the hypothesis that there are no more than t faulty units in an a-t-diagnosable system.

We now consider the case in which $C_0(j) > t$ but $C_0(j)+R_1(i_{jk})$ $> t-k+1$ for each $k = 1,\ldots, C_1(j)$. Under the assumption that the j-th unit in G_2 is faulty, we have now $C_0(j)+1$ faulty units. If the unit corresponding to the row i_{j1} were fault-free, we would have $R_1(i_{j1})-1$ addition faulty units and hence the total number of the faulty units would exceed t. Thus, this unit must be faulty, and we now have $C_0(j) + 2$ faulty units. Then, by the same reasoning, we conclude that the unit corresponding to the row i_{j2} is faulty. The continuation of this argument will lead to the situation where all of the $C_1(j)$ units corresponding to the rows $i_{j1},\ldots,i_{jC_1(j)}$ are faulty. This implies that the number of the faulty units is at least $C_0(j) + C_1(j)+1 = t+1$, a contradiction.

In both of the above cases, we come to a conclusion that the j-th unit in G_2 is fault-free. []

Lemmas 1 and 2 and the statement prior to them lead to the following theorem.

Theorem 2: The diagnosis algorithm correctly locates a fault-free unit in an a-t-diagnosable system in $O(t^2)$ time.

We have just shown that a fault-free unit is always found in the third step of the a-t-d method. The method then uses this unit to test the remaining n-1 units and to identify all faulty units in the system. This, together with the discussion in the second paragraph of this section, leads to the following theorem.

Theorem 3: Let S be a system consisting of n units in which each unit can test any other unit. If S is an a-t-diagnosable system, all faulty units can be identified by at most $n+t^2+t-1$ tests.

The number of tests required in the a-t-d method is at most $n+t^2+t-1$, while that in any deterministic system diagnosis is at least nt. Therefore, adaptive system diagnosis is more desirable than determine system diagnosis in a situation where the cost of a test is a dominant factor.

5. A CONSTRUCTION FOR d-t-DIAGNOSABLE SYSTEMS

Basically, the a-t-d method first performs $t(t+1)$ tests of the t+1 units in G_2 by the t units in G_1, and always locates a fault-free unit in G_2. Then the method uses this fault-free unit to test the remaining n-1 units in a system. Therefore, if each of the t+1 units in G_2 had tested the remaining n-1 units in the system, we would immediately know the faulty units after locating a fault-free unit in G_2. This leads to the following connection assignment of test links for d-t-diagnosable systems.

Let n and t be two positive integers with $n \geq 2t+1$. Let $J=\{1,\ldots,n\}$, $J_1=\{1,\ldots,t\}$ and $J_2=\{t+1,\ldots,2t+1\}$ be sets of n, t and t+1 integers, respectively. A system S of n units, $u_1, u_2, \ldots, u_n$, represented by $G = (U,T)$, is constructed as follows:

$$U = \{u_1, u_2, \ldots, u_n\}, \text{ and}$$
$$T = \{(u_i,u_j)\,|\,i \in J_1 \text{ and } j \in J_2\} \cup \left(\bigcup_{j \in J_2} \{(u_j,u_k)\,|\,k \in J-\{j\}\}\right).$$

It is easily seen from the above discussion that S is a d-t-diagnosable system. The number, $|T|$, of test links in S is $(t+1)(n+t+1)$, and hence S is not an optimal d-t-diagnosable system in the sense that it contains more than nt test links. However, as is evident from the previous section, it essentially takes $O(t^2)$ time to analyze the test results and to identify all faulty units in S. This is superior to the best time complexity O(nt), among the known diagnosis algorithms for d-t-diagnosable systems.

6. MORE ON ADAPTIVE SYSTEM DIAGNOSIS

We begin this section by considering the same example as in Section 3 again. We have concluded from the matrix of test outcomes shown in Fig.1 that units u_4 and u_6 are fault-free. We further claim that u_7 is a fault-free unit and that units u_2 and u_3 are faulty.

We first note that u_2 is a faulty unit; for unit u_2 evaluates a fault-free unit u_6 to be faulty. Suppose that unit u_7 is faulty. Then unit u_1 must be faulty, and hence we already have t = 3 faulty units u_1, u_2 and u_7. Thus, u_3 must be a fault-free unit, which implies that unit u_5 is faulty, a contradiction. Therefore, u_7 is a fault-free unit, and hence u_3 is a faulty unit.

In total, we have located three fault-free units u_4, u_6, and u_7 and identified two faulty units u_2 and u_3, by just analyzing the t(t+1) test results.

We now consider another example. We use the same system S of n=10 units as above in which there are at most t = 3 faulty units. Suppose we obtain the following partial test outcomes.

$$\begin{array}{c|cccc} & u_4 & u_5 & u_6 & u_7 \\ \hline u_1 & 1 & 1 & 1 & 1 \end{array}$$

We can immediately conclude theat u_1 is a faulty unit; for otherwise we have t+1 faulty units. The problem now reduces to the one of identifying at most t-1 = 2 faulty units. Therefore, we only need a new group G_1' of t-1 = 2 units, say u_2 and u_3, and another new group G_2' of t-1+1 = 3 units, say u_4, u_5 and u_6. The units in G_1' start testing the units in G_2'. Suppose we obtain the following test results.

$$\begin{array}{c|ccc} & u_4 & u_5 & u_6 \\ \hline u_2 & 1 & 1 & 0 \\ u_3 & 1 & 0 & 1 \end{array}$$

We can easily conclude that units u_5 and u_6 are fault-free. This implies that u_2 and u_3 are faulty units.

The above examples show that in adaptive system diagnosis, the number of tests required is usually less than $n+t^2+t-1$.

7. CONCLUSIONS

In this paper, we have introduced a new approach to system diagnosis, and shown that this adaptive system diagnosis requires fewer tests and a simpler diagnosis algorithm than the deterministic system diagnosis introduced by Preparata, Metze and Chien [1]. In particular, we have given an upper bound on the number of tests required in the new approach. As illustrated in the previous section, we can usually identify all faulty units by a smaller number of tests. Therefore, finding a lower bound on the number of tests is an interesting problem.

We have demonstrated our new approach using the PMC model. There are several other models [11-13] which have been studied for deterministic system diagnosis. Applying our new approach to these models is another interesting problem.

Recently, Barsi et al. [14] considered the fault diagnosis problem for digital systems using a different test invalidation assumption. This assumption is also fitted into the analog fault analysis problem considered by Wu et al. [9]. Therefore, a third future research direction is to develop a new theory of adaptive system diagnosis with this test invalidation assumption for the analog fault analysis problem.

We have obtained preliminary results in the above research areas. We will report these results in forthcoming papers.

REFERENCES

1. F. P. Preparata, G. Metze, and R. T. Chien, "On the Connection Assignment Problem of Diagnosable Systems," IEEE Trans. on Electronic Computers, Vol. EC-16, pp. 847-854, Dec. 1967.

2. S. L. Hakimi and A. T. Amin, "Characterization of Connection Assignment of Diagnosable Systems," IEEE Trans. on Computers, Vol. C-23, pp. 86-88, Jan. 1974.

3. K. Y. Chwa and S. L. Hakimi, "On Fault Identification in Diagnosable Systems," IEEE Trans. on Computers, Vol. C-30, pp. 414-422, June 1981.

4. A. M. Corluhan and S. L. Hakimi, "On an Algorithm for Identifying Faults in a t-Diagnosable System," in Proc. 1976 Conf. on Information Sciences and Systems, The Johns Hopkins Univ., Baltimore, MD, April 1976, pp. 370-375.

5. T. Kameda, S. Toida, and F. J. Allan, "A Diagnosing Algorithm for Networks," Information and Control, Vol. 29, pp. 141-148, Oct. 1975.

6. R. F. Madden, "An Improved Bound for Fault Identification in t-Diagnosable Systems," Sci. Inst., Univ. of Iceland, unpublished notes, Oct. 1978.

7. G. G. L. Meyer, "A Fault Diagnosis Algorithm for Asymmetric Modular Architectures," IEEE Trans. on Computers, Vol. C-30, pp. 81-83, Jan. 1981.

8. G. G. L. Meyer and G. M. Masson, "An Efficient Fault Diagnosis Algorithm for Symmetric Multiple Processor Architectures," IEEE Trans. on Computers, Vol. C-27, pp. 1059-1063, Nov. 1978.

9. C.-c. Wu, K. Nakajima, C.-l. Wey, and R. Saeks, "Analog Fault Diagnosis with Failure Bounds," in Proc. 24th Midwest Symp. on Circuits and Systems, Univ. of New Mexico, Albuquerque, NM, June 1981, to appear. The extended version, submitted for publication in IEEE Trans. on Circuits and Systems.

10. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.

11. J. D. Russel and C. R. Kime, "System Fault Diagnosis: Masking, Exposure, and Diagnosability without Repair," IEEE Trans. on Computers, Vol. C-24, pp. 1155-1161, Dec. 1975.

12. A. T. Amin, unpublished notes, Bell Laboratories, 1980.

13. S. L. Hakimi, private communication, 1981.

14. F. Barsi, F. Grandoni, and P. Maestrini, "A Theory of Diagnosability of Digital Ssytems," IEEE Trans. on Computers, Vol. C-25, pp. 585-593, June 1976.

ON TESTING STUCK-OPEN FAULTS IN CMOS COMBINATIONAL CIRCUITS

R. CHANDRAMOULI
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA. 91109

ABSTRACT

Recently it has been found that a class of failure related to a particular technology (CMOS) cannot be modelled as the conventional stuck-at fault model. These failures change the combinational behaviour of CMOS logic gates into a sequential one. Such a failure is modelled as a fault, called the Stuck-Open fault (SOP).

The object of this paper is to develop a procedure to detect single SOPs in combinational circuits. It is shown, that in general, tests generated for stuck-at faults when applied in a particular sequence will detect all single SOP faults. In case of single redundancy in the network, the SOP fault on the redundant line cannot be detected. When there is reconvergent fan-out in the network, there is a one-one correspondence between the conditions for stuck-at fault and stuck-open fault detectability.

INTRODUCTION

Over the past decade there has been an increased use of MOS technology in the implementation of LSI/VLSI devices and specifically the emergence of CMOS technology in low power applications because of its low power dissipation characteristics [1]. This higher level of integration has associated with it the problem of testing such devices. The stuck-at fault model that was used in earlier works [2], [3], [4], [5] is found to be inadequate in defining many physical failures. Recently Wadsack [6] developed a fault model for certain types of failures called stuck-open faults in CMOS logic gates. The behaviour of a stuck-open fault is different from that of a stuck-at fault, in that it causes the logic gates to behave like a memory element; i.e. the combinational element behaving like a sequential element. Hence such failures cannot be tested by the conventional combinational testing approach.

Since CMOS will emerge as a leading process in low power, high density, VLSI devices in the coming decade, testing these devices for stuck-open faults becomes a recognized requirement. Although Wadsack has modeled the stuck-open fault in his paper [6], it doesn't deal with the effect of stuck-open faults in a combinational circuit as well as an effective procedure for testing them in combinational circuits.

The object of this study is to develop a test generation approach for detecting single stuck-open faults in CMOS logic circuits, and analyse the effect of redundancy and reconvergent fan-out on testing stuck-open faults.

STUCK-OPEN FAULT (SOP) MODEL

Figure 1(a) shows a 2-input NAND gate and 1(b), its implementation using CMOS technology. The x's in Figure 1(b) corresponds to "open" conditions, which are electrically equivalent to missing connections between two points.

Assume a single "open" condition at "d" (henceforth single "open" condition at "d" will be referred to simply as "d"). When (00) is applied to inputs (AB), the n-transistors are turned off and the p-transistor corresponding to A does not conduct because of "d." The p-transistor corresponding to B will, however, conduct, and the output will be 1. This corresponds to normal (fault-free) NAND gate response. If on the other hand a (01) is applied to (AB), the n-transistor chain will be turned off as before, as well as the p-transistor corresponding to input B because of

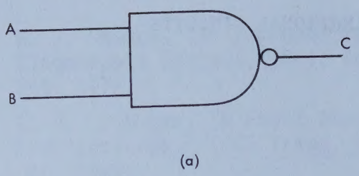


Figure 1(a). 2-Input NAND gate.

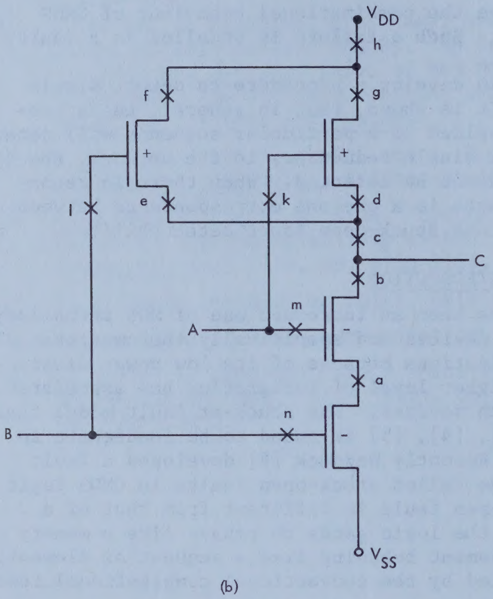


Figure 1(b). CMOS NAND gate with "open" (X) conditions.

a 1 at its input. Under fault-free conditions, the p-transistor corresponding to A should have been turned on with a 0 at its input resulting in the output C pulled to logic 1 state. However, the presence of "d" prevents the charging of the output node through the p-transistor. Hence the output node, because of the predominantly capacitive load effect, retains the logic value of the previous output state, and the retention time depends on the leakage current at the output node. Thus, for a certain input condition, the current output value depends on the previous output at the output node, under a single "open" condition. In short, the logic gate (combinational) behaves like a memory element (sequential) in the presence of an "open" failure. Table 1 summarizes the output response of the gate to all input combinations under all single "open" conditions. Q^n in the table corresponds to previous output value synonymous with sequential circuit terminology.

Summarizing, the equivalent sets of faults (e,f,l) and (d,g,k) are due to faulty pull-ups (p-transistor) and (a,b,m,n,) are due to faulty pull-downs (n-transistor). Hence the definition:

Definition: A fault due to a faulty pull up or pull down in a CMOS logic (Combinational) device causing it to behave like a memory element (Sequential) is defined as "Stuck-open Fault" (SOP).

(d,g,k) and (e,f,l) are defined as input stuck-open faults (ISOP) and (a,b,m,n) are defined as output stuck-open faults (OSOP).

Table 1. Truth Table for Normal and Faulty NAND Gate

			C_f											
A	B	C	a	b	c	d	e	f	g	h	k	l	m	n
0	0	1	1	1	Q^n	1	1	1	1	Q^n	1	1	1	1
0	1	1	1	1	Q^n	Q^n	1	1	Q^n	Q^n	Q^n	1	1	1
1	0	1	1	1	Q^n	1	Q^n	Q^n	1	Q^n	1	Q^n	1	1
1	1	0	Q^n	Q^n	0	0	0	0	0	0	0	0	Q^n.	Q^n

C_f = Output under single fault condition
C = Fault-free output

Since SOP's cause combinational behaviour to fail into sequential behaviour, they have to be tested using a sequential circuit testing approach. One of the approaches is to reset the given circuit to a known state using either a synchronizing or homing sequence followed by sequencing the circuit through all possible states and input transisions [7]. In an analogous manner, to test for an ISOP in a NAND gate, the gate output is discharged (logic 0) first (equivalent to resetting the output to a known state) and then is charged through the faulty pull up by applying 0 to the faulty input and 1 to the other input. An OSOP is tested similarly, first by charging the gate followed by discharging it. Table 2(a), (b) and (c) lists the SOP tests and the corresponding normal (C) and faulty (C_f) response of the NAND gate. t_0 and t_1 in Table 2 correspond to time periods and $t_1 > t_0$.

Table 2(a). Test for A ISOP, (b) Test for B ISOP, (c) Test for C OSOP

	t_0	t_1
A	1	0
B	1	1
C/C_f	0/0	1/0

(a)

	t_0	t_1
A	1	1
B	1	0
C/C_f	0/0	1/0

(b)

	t_0	t_1
A	0	1
B	0	1
C/C_f	1/1	0/1

(c)

A similar analysis can be done for NOR gates, and the results will be dual of NAND gate results.

SINGLE SOP FAULT DETECTION IN CMOS COMBINATIONAL CIRCUIT

Irredundant Non Reconvergent Fan-Out Network (NRFO)

First, the effect of single SOP's in an irredundant NRFO will be examined. The analysis will be restricted to NAND type networks which can be very easily extended to other gate types.

$I_k = \{I_0, I_1\}$ is an ISOP test for a NAND gate input in the path P_k of the network. I_0 and I_1 are applied at time t_0 and t_1 respectively to the NAND gate and $t_1 > t_0$.

From Table 2, it is evident that I_k for one of the inputs of an n-input NAND gate will be {(111...1...1), (111...0...1)}

T^1 is the set of all ISOP tests for all primary inputs (PI) in the network.

T^2 is the set of all ISOP tests for all level 1 gate output lines in the network.

Lemma 1: In an irredundant NRFO network T^1 and T^2 together will detect all single ISOP's in every component of all the paths in the network.

Proof: Let P_i = (1,2,...,n.n+1) represent a path from the primary input (PI) line 1 to the primary output line n+1 in Fig. 2. (1,2,...,n+1) are called the components of the path P_i and each component corresponds to the input of a gate in the network. (NAND network is assumed here.)

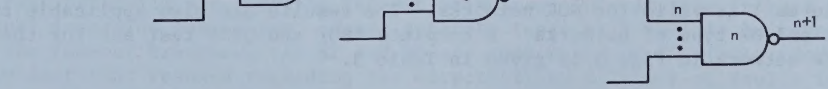


Figure 2. n-Level NRFO NAND Network

Consider an ISOP test for line 1 in Figure 2. I_k is applied to gate 1 and inputs to other gates are assigned such that the fault in line 1 is propagated to the primary output (PO). $I_k = \{(11...1), (011...1)\}$ applied to gate 1 will generate {0,1} at line 2 which is a component of the path P_i. Since the test sensitizes the path P_i, all the gates in this path will have their inputs which are not the components of the path P_i set at logical 1 [3]. Hence the inputs (components of the path) to every gate along the path P_i will be inverted. This is because, a NAND gate with all except one of its input tied at logic 1 inverts the input which is not tied to 1. The {0,1} sequence at line 2 of P_i (due to I_k at line 1) will be inverted to {1,0} and again inverted to {0,1} by gate 3. Proceeding in this fashion, it can be seen that every odd component (with respect to line 1) of the path P_i will have a {1,0} applied to it. Since {1,0} at all odd components of this path and {(11...1),(11...1)} at other inputs of the gates in the path correspond to ISOP tests for the odd components of the path P_i, the test for line 1 tests for single ISOP in all odd components in the path. Similarly an ISOP test for line 2 will correspond to the application of {(011...1),(11...1)} to gate 1 which will generate {1,0} at line 2 and 1's to all other inputs of gate 2. In a similar manner the ISOP test for line 2 will generate {1,0} at every even component of the path (with respect to line 1). Since this is a sensitized path, {1,0} at every even component of the path represents an ISOP test for each of these components in the path P_i. As T^1 corresponds to ISOP tests for all PI's, it will detect all single ISOP's in all odd components of all paths from the PI's. Similarly T^2 corresponds to tests for lines, which are outputs of gates at level 1, and hence will detect single faults in all even components of all paths in the network. Hence the lemma is proved.

Q.E.D.

Since (11...1) is a stuck-at-Zero test (SZ) and (011...1) is a stuck-at-one-test (SO) for a NAND gate input, an ISOP test for line 1 is equivalent to {(SZ),(SO)}. In the above case, in order to generate an ISOP test for line 2 we have to apply {(01...1),(11...1)} to gate 1 and 1's to all other inputs of gate 2 except line 2. This is the same as {(SO),(SZ)}. Hence T^1 corresponds to {(SZ),(SO)} to be applied to every PI and T^2 corresponds to {(SO),(SZ)} to be applied to every PI. Since T^1 and T^2 are both needed to test for all single ISOP's, we need to apply both {(SZ),(SO)} and {(SO),(SZ)} to every primary input in the network (in case of primary input fan-out, only the fan-out branches are considered). However, {(SZ),(SO)} and {(SO),(SZ)} can be combined together such that (SO) of T^1 and (SO) of T^2 overlap. This produces a shorter sequence, {(SZ),(SO),(SZ)}, where (SZ) is applied at t_0, (SO) applied at t_1 and finally (SZ) applied again at t_2 ($t_2 > t_1 > t_0$).

Since this sequence applied to every PI in a gate tests the gate output line more than once, it will be sufficient to apply {(SZ),(SO),(SZ)} to one PI of every gate and {(SZ),(SO)} to other PI's of that gate. From Table 2 we can see that {(SO),(SZ)} will test for an OSOP and hence {(SZ),(SO),(SZ)} will also detect OSOP in each gate in the network. This leads to the following theorem:

Theorem 1: All single ISOP's and OSOP's in an irredundant NRFO network can be detected by applying the sequence {(SZ),(SO),(SZ)*} to every PI in the network except in the case of (SZ)* which is applied to only one PI of every gate in level 1.

Since NOR networks are the dual of the NAND networks, the dual of Theorem 1 is valid for NOR networks. The results are also applicable to AND and OR type of networks. A complete ISOP and OSOP test set for the NRFO network in Fig. 3 is given in Table 3.

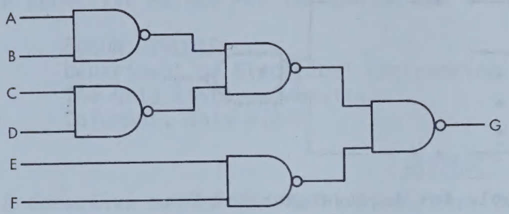


Figure 3. NRFO Example Network

Table 3. SOP Test Set for Network of Figure 3

Input	t_0	t_1	t_2
A	SZ	SO	SZ
B	SZ	SO	
C	SZ	SO	SZ
D	SZ	SO	
E	SZ	SO	SZ
F	SZ	SO	

Redundant Networks

A network is redundant if removal of lines or gates in the network does not change the primary output function. A single redundancy in an NRFO network corresponds to an undetectable stuck-at fault [8]. This can be seen in Fig. 4, where the stuck-at fault at X cannot be detected. Thus we do not have tests for the stuck-at fault at C. Hence the following lemma,

Lemma 2: A single redundancy in a NAND or NOR network will render the ISOP fault on the corresponding line undetectable.

As in the case of single redundancy, ISOP faults on lines which are jointly redundant are jointly undetectable.

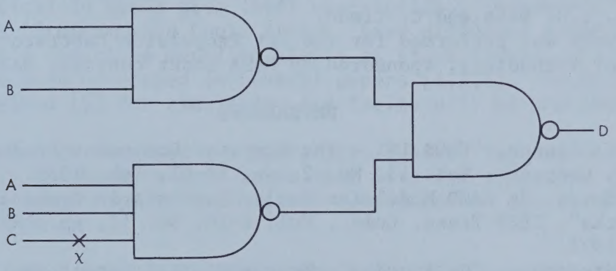


Figure 4. Redundant Circuit

Reconvergent Fan-out Networks (RFO)

In RFO networks a signal can travel in multiple paths where all paths can have either equal or unequal inversion parities. In case of equal inversion parity, stuck-at faults at the fan-out origin can be detected by propagating its effect through all paths and in case of unequal inversion parity, cannot be propagated simultaneously through all paths [3]. An example of a reconvergent fan-out situation is shown in Fig. 5. Unlike in a stuck-at fault case, a stuck-open fault is associated only with the fan-out branches and not the fan-out stem. Since any stuck-open fault in a path preceding the fan-out stem has to be propagated through all fan-out branches (equal inversion parity along the fan-out branches), the ISOP test for that path will not test the fan-out branches. This is true in case of unequal inversion parity too. Hence separate tests will have to be generated for the fan-out branches. As SZ and SO are components of SOP fault test, it is evident that results regarding the detectability of stuck-at faults in RFO networks will have one-one correspondence with the detectability of SOP faults.

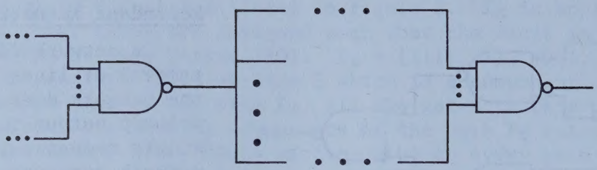


Figure 5. An example for Reconvergent Fan-Out

CONCLUSION

CMOS is in the forefront of VLSI technology and hence an analysis of a failure mode peculiar to CMOS and methods to test them is highly desirable. Since none is available currently, this study analyses the effect of stuck-open faults, a class of failure in CMOS combinational circuits and develops a procedure for testing them. Results derived show that no special test generation procedure is required and tests developed for stuck-at faults if applied in a given order will detect all single stuck-open faults in the network. It is worth noting that this procedure requires very little extra effort in terms of test sequence generation and actual testing process. In case of redundancy and reconvergent fan-out, SOP fault detectability conditions are the same as in the case of stuck-at faults.

ACKNOWLEDGEMENTS

The author acknowledges the managerial and technical assistance of W.R. Scott, L.M. Hess and C. Timoc.

This work was performed for the Jet Propulsion Laboratory, California Institute of Technology, sponsored by NASA under contract NAS7-100.

REFERENCES

1. D.L. Wallensen, "CMOS LSI - The Computer Component Process of the 80's", Computer, Vol. 13, No. 2, pp. 59-67, Feb. 1980.
2. J.P. Hayes, "A NAND Model for Fault Diagnosis in Combinational Logic Networks", IEEE Trans. Comp., Vol. C-20, No. 12, pp. 1496-1506, Dec. 1971.
3. D.B. Armstrong, "On Finding a Nearly Minimal Set of Fault Detection Tests for Combinational Logic Nets", IEEE Trans. Elect. Comp., Vol. EC-15, pp. 66-73, Feb. 1966.
4. J.P. Poage, "Derivation of Optimum Tests to Detect Faults in Combinational Circuits", in Proc. Symp. Math. Th. Automata, pp. 485-528, Apr. 1962.
5. J.P. Roth, "Diagnosis of Automata Failures: A Calculus and a Method, "IBM J. Res and Dev., Vol. 10, pp. 278-291, July 1966.
6. R.L. Wadsack, "Fault Modelling and Simulation of CMOS and MOS Integrated Circuits", The Bell System Tech. Journ., Vol. 57, pp. 1449-1474, June 1978.
7. F.C. Hennie, "Fault Detection Experiments for Sequential Circuits", Proc. Fifth Ann. Symp. Swit, Circ. Th. and Logic Design. pp. 95-110, Nov. 1964.
8. H.S. Lee, "Redundancy Testing in Combinational Networks", IEEE Trans. Comput., Vol. C-23, No. 10, pp. 1029-1047, Oct. 1974.

A DEDUCTIVE METHOD FOR THE SIMULATION OF FAULTS IN PROGRAMMABLE LOGIC ARRAYS

FUSUN OZGUNER
Department of Electrical Engineering
The Ohio State University
Columbus, Ohio 43210

ABSTRACT

A deductive method for fault simulation in Programmable Logic Arrays (PLA's) is presented. By taking into consideration the structural properties of a PLA, the output fault lists can be constructed by observing the output values or by simple list operations. It is also shown that fault list calculations in an inverterfree circuit can be done by performing fewer set operations compared to methods for random logic.

INTRODUCTION

Fault simulation is becoming an increasingly important part of fault diagnosis of digital systems, as the circuits become larger and test generation becomes time consuming and expensive. Several effective algorithms for the simulation of stuck type faults exist in the literature; i.e., parallel, deductive and concurrent simulation. The parallel and deductive methods have been compared in [6]. It has been shown that the deductive method is faster.

In LSI and VLSI circuits gate level descriptions may not be available or it may be impractical to use a gate level description. Components like memories and PLA's (Programmable Logic Arrays) have different error modes and defects in PLA's cannot be modelled simply as stuck faults. Fault models for PLA's have been developed in several papers [1,2,3]. In this paper, a deductive method [5] for simulating PLA faults will be presented.

Deductive simulation involves set operations (union, intersection) on fault lists. It will be shown in this paper that list operations on fault lists can be greatly simplified by taking into consideration the structural properties of PLA's.

GATE LEVEL DEDUCTIVE SIMULATION OF INVERTERFREE CIRCUITS

List operations in deductive simulation take a considerable amount of computation time. In the gate level conventional deductive simulation, the output fault list for each of the gates is computed in the following way. The set operations performed on the fault lists of the gate inputs depends on the gate input and output values. To summarize the principles let's consider an AND gate. Similar computations are done for other types of gates.

1. The output of the AND gate is 1.

   This means that all the inputs are 1. Any fault which changes an input to a 0 will change the output to 0 and should be on the output fault list. So, the fault list of the output is the union of the fault lists of the inputs.

2. The output of the AND gate is 0.

   At least one input must be 0. A fault in the previous levels can change the output of the gate to a 1 only if it changes all the 0 inputs to 1 and does not change any 1 input to 0. Therefore, the fault list of the output is obtained by taking the intersection of the fault lists of the 0 inputs and subtracting the fault list for each 1 input from this intersection.[1]

The faults on the inputs or output of the gate that is being processed, that would change the output of the gate must also be added to the fault list. For example, in the case of an AND gate whose output is 1, the stuck-at-0 fault on its output would be added to its output fault list.

Fewer list operations need to be performed if the simulated circuit is inverterfree.

Theorem 1: In an inverterfree circuit fault lists of lines with complementary logic values are disjoint.

Proof: The proof follows from the fact that a s-a-1 (s-a-0) fault can only cause erroneous 1's (0's) at gate inputs and outputs on paths from the fault site to the circuit outputs. The fault list of a line with value 1(0) consists of faults that would change it to a 0(1) and in this case these could only be s-a-0 (s-a-1) faults. Therefore, the same fault could not appear on the fault lists of lines with complementary logic values.

This means that in fault simulation of an inverterfree circuit by the deductive method, the same fault could not appear on the fault lists of inputs to the same gate with different logic values, and therefore the set operations performed for the calculations of the gate output fault list are simplified. Notice that there can be reconvergent fanout in the circuit.

In the conventional deductive simulation algorithm, the output fault list L_{out} of an AND gate with output value 0 in the faultfree circuit is calculated as follows:

$$L_{out} = L_1^0 \cap L_2^0 \cap \dots . L_i^0 \cap \dots \overline{L_1^1} \cap \dots \overline{L_j^1} \cap \dots \qquad (1)$$

where L_i^0 = fault list of the ith input with value 0

and L_j^1 = fault list of the jth input with value 1.

By Theorem 1 $L_i^0 \cap L_j^1 = \emptyset$ in an inverterfree circuit and it can easily be seen that $L_i^0 \cap \overline{L_j^1} = L_i^0$. Therefore (1) simplifies to

$$L_{out} = L_i^0 \cap L_i^0 \cap \dots \cap L_n^0$$

$$\text{or} \quad L_{out} = \bigcap_{i=1} L_i^0 \qquad (2)$$

where n is the number of 0 inputs of the AND gate.

As an example consider the AND gate in Fig. 1. Suppose that this AND gate is in an inverterfree circuit. The inputs of the gate have the values A=0 B=0 C=1 D=1 and its output is 0. The fault lists of the gate inputs are

$L_A = \{2, 5, 6, 12\}$

$L_B = \{1, 5, 9\}$

$L_C = \{3, 4, 8\}$

$L_D = \{3, 10\}$

---

[1] Subtraction is defined as $A-B = A \cap \overline{B}$

Since the effects of faults in the inverterfree circuit are unidirectional

$$L_A \cap L_C = \phi, \quad L_A \cap L_D = \phi, \quad L_B \cap L_C = \phi \quad \text{and } L_B \cap L_D = \phi$$

Using conventional deductive simulation L_{out} would be calculated in the following way:

$$L_{out} = L_A \cap L_B \cap \bar{L}_C \cap \bar{L}_D$$

$$= \{5\}$$

Using Theorem 1 the same result is obtained by performing fewer set operations.

$$L_{out} = L_A \cap L_B$$

$$= \{5\}$$

One important application of this result is the deductive simulation of stuck faults of a PLA as will be seen in the next section.

DEDUCTIVE SIMULATION OF FAULTS IN PROGRAMMABLE LOGIC ARRAYS

PLA faults can be modeled as: [1,2,3]

1) crosspoint faults
2) stuck faults
3) shorts

Stuck faults are the classical stuck faults in the equivalent AND-OR circuit. Crosspoint faults are due to errors in the process of programming a PLA. A missing device (MD) fault is the absence of a contact where there should be one. An extra device (ED) fault is an additional contact where there should not be a contact. We will not be considering short faults here, because of the large number of possible shorts. A PLA is shown in Fig. 2. In the following discussions, a decoder output can mean an input variable, its complement or a term generated by the decoder outputs. Since each output of the AND array corresponds to a product term we will be referring to product term fault lists throughout the discussions.

Calculation of Crosspoint Fault Lists of the AND Array

An MD fault in the AND array is equivalent to a s-a-1 fault of the AND input in the equivalent AND-OR circuit [2]. An ED fault on the other hand, has no equivalent stuck fault. The effect of these two types of faults on the outputs of the AND array can be analyzed by observing the input and output values of the product term as done in the deductive simulation of random logic:

a) For test T, the value of the product term is 0

   i) One input of the product term = 0

   This case is equivalent to the case of an AND gate with one input at value 0 and other inptus at value 1. The MD fault corresponding to the input literal with value 0 will be the only fault in the fault list of the product term, since no ED fault can change the product term to a 1.

ii) Number of 0 inputs of the product term > 1

No crosspoint fault can change output to 1. Therefore, the list is empty.

b) For test T, value of the product term is 1.

An MD fault will not change the value of the product term as one literal will missing and the value will still be 1. An ED will affect the value of the product term if the input literal that is erroneously connected to the product term has value 0 for test T. Therefore, the ED faults at the crosspoint of the 0-valued decoder outputs and the 1-product term should be placed in the fault list of that product term.

Fault List Propagation Through the OR Array and Calculation of Crosspoint Fault Lists of the OR Array

Crosspoint fault lists calculated for the AND array have to be propagated through the OR array, and the crosspoint faults of the OR array that would affect the output values should be added to the fault lists of the output lines. Then the crosspoint fault list L_C of a PLA output is

$$L_C = L_C^P \bigcup L_C^{OR} \tag{3}$$

where

L_C^P: list of propagated crosspoint faults

L_C^{OR}: list of OR array crosspoint faults that change the output

Calculation of L_C^{OR} is similar to that discussed for the AND array. Similar calculations can be made for a NOR-NOR PLA.

Property 1: Crosspoint fault lists of product terms are disjoint.

Using this property, fault list calculations for the OR array outputs can be done by performing fewer set operations compared to deductive simulation of random logic. There is no need to perform intersection and subtraction operations on the fault lists of the product terms. Again, fault list calculation depends on the output value for the faultfree circuit:

a) For test T, PLA output $F_j = 1$

i) One OR input (product term) has value 1

A crosspoint fault can change the output to a 0 only if it changes the 1-product term to a 0. By property 1, crosspoint fault lists of the product terms are disjoint and therefore the same fault cannot change a 1-input to a 0 and a 0-input to a 1, as in random logic. Output fault list calculation for propagated faults can be done as follows:

$$L_C^P = \text{Fault list of the 1-product term}$$

$$L_C^P = \bigcup \text{ED fault at the crosspoint of the 1-product term and the 0-input literals (or decoder outputs)} \qquad (4)$$

If calculated as in random logic, the fault list L_C^{*P} of such an OR gate would be

$$L_C^{*P} = \text{List of 1-input} - (\bigcup \text{Lists of 0-Inputs}) \qquad (5)$$

$$\text{since } L_C^{OR} = \text{MD fault of the 1-product term in the OR array} \qquad (6)$$

$$L_C = \{\bigcup \text{ED faults at the crosspoint of the 1-product term and the 0 decoder outputs}\} \bigcup \text{MD fault of the 1-product term in the OR array.} \qquad (7)$$

ii) More than one OR input has value 1

A fault can change the output F_j to a 0, only if it appears in the lists of all 1-product terms. Since product term lists are disjoint, no single crosspoint fault in the AND array can change F_j to a 0. Therefore,

$$L_C^P = \{\phi\} \qquad (8)$$

If calculated as in random logic

$$*L_C^P = (\bigcap \text{lists of 1-inputs}) - (\bigcup \text{list of 0-inputs}) \qquad (9)$$

Since

$$L_C^{OR} = \{\phi\} \qquad (10)$$

we obtain

$$L_C = \{\phi\} \qquad (11)$$

b) For test T, output $F_j = 0$

$$L_C^P = \bigcup_{\text{inputs}} L_{\text{input}} \qquad (12)$$

Since a 0-product term fault list has at most one MD fault L_C^P is a union of disjoint, single element or empty sets and

$$L_C = \{\bigcup_{(\text{OR inputs } \varepsilon P_0)} \text{MD fault in AND array of 0 input literal of product term}\} \bigcup \{\bigcup \text{ED faults at the crosspoint of 1-product terms and } F_j\} \qquad (13)$$

where

$$P_0 = \text{set of product terms with a single 0 input literal}$$

As seen from the above discussions, the crosspoint fault lists of the PLA outputs can be obtained directly by observing the output values of the PLA.

As in any deductive simulation algorithm, line values for the given test have to be calculated. During this preprocessing, the names or ID numbers of product terms with a single 0-input, and 1-product terms can be stored and used later to construct output fault lists. Then output fault lists can be calculated by observing output values by using (7) or (13).

Simulation of Stuck Faults in a PLA

Stuck faults in a PLA are defined to be the stuck faults in the gate equivalent [2]. Some of these faults are equivalent to missing device crosspoint faults. The s-a-1 fault of an AND input is equivalent to the MD fault of the corresponding product term input. Similarly the s-a-0 fault of an OR input is equivalent to the MD fault at the crosspoint of the corresponding product term and the output line. So, the stuck faults, that should be considered are: 1) decoder output or input literal stuck faults, 2) product term stuck faults (AND outputs) and 3) PLA output stuck faults.

Stuck faults of types 2 and 3 that would be detected by a test pattern can easily be deduced from the PLA output values like the crosspoint faults. If L_S is the stuck fault list for the product term stuck faults and the PLA output stuck faults, then

a) If PLA output $F_j = 0$

$L_S = \{\bigcup_{\text{OR inputs}}$ connected product term s-a-1 fault$\} \bigcup$ output F_j s-a-1 (a connected product term denotes a product term which is used to form the OR function for output F_j) (14)

b) If PLA output $F_j = 1$

i) one connected product term = 1

L_S = s-a-0 fault of connected 1-product term $\bigcup$ output F_j s-a-0 (15)

ii) more than one connected product term = 1

L_S = output F_j s-a-0 (16)

The PLA output fault list L for these faults and crosspoint faults can be obtained from:

$$L = L_C \bigcup L_S \tag{17}$$

Decoder Output Stuck Faults

Stuck faults of the decoder output lines can cause erroneous values on more than one product term and it is possible for several of these erroneous values to reconverge as inputs to the same OR function. However, as a PLA is essentially an AND-OR circuit, the effect of this type of a fault will be unidirectional and Theorem 1 can be used for fault list calculation during deductive simulation

CONCLUSIONS

The simulation of crosspoint and stuck faults in PLA's has been considered. It has been shwon that, because of the regular structure of a PLA, the crosspoint faults and the product term stuck faults that would be detected by a test pattern can be easily determined. As a result of Theorem 1, fault list calculations in an inverterfree circuit can be done by performing fewer set operations compared to random logic. Deductive simulation of decoder output stuck faults is also a consequence of this theorem.

REFERENCES

[1] C.W. Cha; "A Testing Strategy for PLA's", Proc. 15th Design Automation Conference, pp. 326-334, Las Vegas, Nevada, June 1978.

[2] J.E. Smith, "Detection of Faults in Programmable Logic," IEEE Transactions on Computers, Vol. C-28, pp. 845-854, Nov. 1979.

[3] D.L. Ostapko and S.J. Hong, "Fault Analysis and Test Generation for Programmable Logic Arrays (PLA)", Proc. FTCS-8, pp. 82-89, Toulouse, France, June 1978.

[4] V.K. Agarwal, "Multiple Fault Detection in Programmable Logic Arrays," IEEE Transactions on Computers, Vol. C-29, pp. 518-522, June 1980.

[5] D.B. Armstrong, "A Deductive Method for Simulating Faults in Logic Circuits," IEEE Transactions on Computers, Vol. C-21, pp. 464-471, May 1972.

[6] F. Ozguner, C.W. Cha and W.E. Donath, "On Fault Simulation Techniques," Journal of Design Automation and Fault Tolerant Computing, pp. 83-92, April 1979.

[7] S.J. Hong, "Fault Simulation Strategy for Combinational Logic Networks," Proceedings of the Eighth Annual International Conference on Fault-Tolerant Computing, pp. 96-99, Toulouse, France, June 1978.

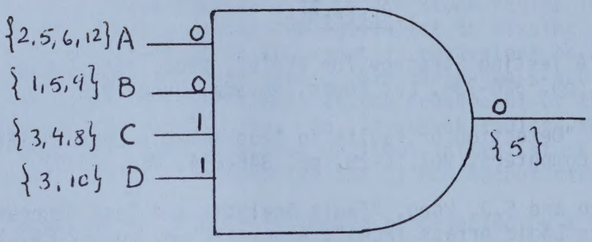

{2,5,6,12} A
0
{1,5,9} B
0
{3,4,8} C
1
{3,10} D
1
0
f
{5}

Figure 1 Fault List Calculation for an AND Gate in an Inverterfree Circuit

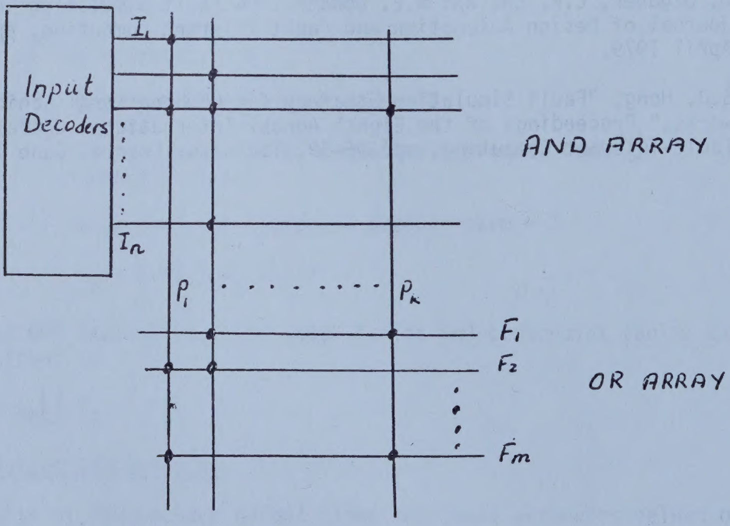

I₁
Input Decoders
Iₙ
AND ARRAY
P₁
Pₖ
F₁
F₂
Fₘ
OR ARRAY

Figure 2 Programmable Logic Array

MORPHIC PROPERTIES OF DETERMINISTIC AND NONDETERMINISTIC FAULT MODELS[1]

B. L. Havlicsek and G. G. L. Meyer
Electrical Engineering Department
The Johns Hopkins University
Baltimore, Maryland 21218

ABSTRACT

The model described in this paper [2] permits the analysis of faults for a wide variety of digital, analog and hybrid systems. By allowing the association of more than one syndrome with a given fault situation and the consideration of test outcomes other than pass and fail, it is possible for this model to represent structures present at high levels of complexity. In particular, this work examines the possible relationships which may exist between the syndromes associated with a multiple fault situation and the syndromes associated with the components of that fault situation. It is shown that only a small number of relations are possible and that these include the relations which correspond to the high level fault models of Sogomonyan [4], Preparata, Metze and Chien [3], and Barsi, Grandoni and Maestrini [1].

REFERENCES

[1] Barsi, F., Grandoni, F. and Maestrini, P., A Theory of Diagnosability of Digital Systems, IEEE Transactions on Computers, Vol. C-25, June 1976, pp. 585-593.

[2] B. L. Havlicsek and G. G. L. Meyer, "Morphic Properties of Deterministic and Nondeterministic Fault Models", Johns Hopkins University, Electrical Engineering Department, Technical Report JHU-EE-81-15, 1981.

[3] Preparata, F. P., Metze, G. and Chien, R. T., On the Connection Assignment Problem of Diagnosable Systems, IEEE Transactions on Electronic Computers, Vol. EC-16, December 1967, pp. 848-854.

[4] Sogomonyan, E. S., Monitoring Operability and Finding Failures in Functionally Connected Systems, Automation and Remote Control, Vol. 25, No. 6, June 1964, pp. 874-882.

---

[1] This work was supported in part by the Office of Naval Research, under Contract N00014-80-C-0772 and in part by the Westinghouse Electric Corp., Integrated Logistics Support Division, Hunt Valley, MD.

MODELING AND RELIABILITY OF MULTI-MINIPROCESSOR SYSTEMS

KONSTANTINOS KONSTANTINIDES
Systems Science Department - UCLA
Los Angeles, California 90024

ABSTRACT

In this work we develop an analytical model of a system with an arbitrary number of CPUs and I/Os.

We assume that the CPUs are not reliable and that we can fix only one at a time. We assume also that all times are independent and exponential, and all disciplines are F.C.F.S.

We compare the throughput of a single-CPU system, (with service rate 'μ'), with the throughput of a multi-miniprocessor system with N CPUs, where now each CPU has service rate μ/N.

Finally we compare our results with those of an approximate solution, given by C. Sauer and K.M. Chandy.

I. INTRODUCTION

As it is already known, the decreasing cost of multi-miniprocessor systems made them an alternative to conventional architectures.

Hence, it worths to study, if the performance (in terms of reliability, mean queue length, throughput and utilization) of a computer system with one CPU, with service rate 'μ', is better or worse than the performance of a computer system with N CPUs, where now each CPU has service rate μ/N, and how we can increase the performance of a multi-miniprocessor system.

In this work we try to answer those problems for some special cases, presenting a new analytical model of a multi-processor system.

Browne et al. [1] gave some results, but in their work they assume that all CPUs are reliable.

Here we assume, (in both the single-CPU and N-CPU systems), that our CPUs are unreliable. They fail with a rate 'γ' , maybe more than one at a time, but we can fix only one at a time with a rate 'α'. All the I/Os are reliable and they have a service rate 'λ'. All times are exponential and all disciplines are First Come First Serve. (F.C.F.S.).

Also we assume that both the single CPU and N-CPU systems have the same number of I/Os, (N2), and that there is a maximum number of programs in the systems, M.

We use a cyclic model (Fig. 1) to model the system, where all I/O devices share a common queue. In order to solve the Markov-Chain equations we use a common used recursive technique. [2,3]

Sauer and Chady gave also an approximate solution to the same problem, [4] therefore we begin by describing their model in section II. In section III we describe our model and how we can use it to calculate different performance parameters of a multi-processor system, like throughput, utilization, and mean queue length of the CPUs. In section IV we present results we got by using our model, and we compare them with results we got by using the model by Sauer and Chandy. Finally we end with our conclusions.

II. A MODEL BY SAUER AND CHANDY

Sauer and Chandy used a simplified approach to model an unreliable multi-miniprocessor system.

They introduced the term availability (AV), which they defined as:

$$AV = \frac{\alpha}{\alpha + \gamma}$$

where 'α' is the rate of repairs
and 'γ' is the rate of failures.

For the single-CPU system they use the Markov-model of Fig. 2. In that model we have two states.

State (1): The CPU works

State (0): The CPU does not work.

With simple calculations we have that P(1) = AV.

Now, if the throughput of the system, when AV = 1, is equal to THRS(1), then the long term average throughput of the system will be:

$$THRS1=THRS(1)*AV$$

Now, suppose we have N CPUs. Here we can use the Markov model of Fig. 3, where we have N+1 states, and each state shows how many processors work. We define: P(i)=Pr(i processors work), then if THR(i) is the throughput of a multi-processor system with i processors and with AV=1, they calculate the throughput of the system as:

$$THRM = \sum_{i=1}^{N} P(i)*THR(i)$$

As we can see, there might be a problem with this method, since the authors use only one unknown parameter in their solution, (the availability), while in fact we have two parameters, ('α' and 'γ'). We can see that for the same value of the availability, the parameters 'α' and 'γ' can take an arbitrary number of values. Therefore, the whole performance of the system may change in an important way.

This fact made us introduce an analytical model, in order to solve the same problem and so to examine the validity of the results we had by using the Sauer and Chandy model.

III. AN ANALYTICAL METHOD

Again, as before, we have to model a multiprocessor system, where the CPUs fail at a rate 'γ' and we can fix only one at a time with a rate 'α'. All times are exponential and all disciplines are F.C.F.S.

Our goal is to make also, a computer program as much general as possible, which will be able to calculate various performance characteristics of this system that has an arbitrary number of CPUs and I/Os. Such characteristics could be: the throughput of the system, the mean queue length, or the utilization of the CPUs.

In order to do that, we give first a full diagram of the Markov states of the system. (Fig. 4). State (i,j) denotes that there are i programs in the CPU and that j CPUs are working.

Let P(i,j) be the equilibrium probability of state (i,j). In order to solve the Markov balance equations we use a recursive technique. Hence our algorithm solves only for the state probabilities P(0,1), P(0,2),....P(0,N) and the other state probabilities are obtained recursively. Performance metrics can also be determined in terms of those probabilities. For more details about this method one can refer to {2} and {3}.

Algorithm

1. Initialization. All state probabilities are treated as N-element column vectors until the end of the algorithm. The vector p(i,j) corresponds to P(i,j). (So in a fortran program all p(i,j)s are elements of a three-dimensional array.)

$$p(0,N) = (1,0,..,0)^T$$
$$p(0, N-1) = (0,1,0,.,0)^T$$
....
$$p(0,1) = (0,...,0,1)^T$$

2. Solve for $p(0,0)$
Use the vector g to sum all the probabilities.

$$p(0,0) = \frac{p(0,1)\gamma}{\alpha + \lambda\beta\ (N2,M)}$$

3. Solve for $p(1,N)$.

$p(1,N) = \{p(0,N)(\lambda' + N\gamma) - p(0,N-1)\alpha\}/\mu\beta(1,N) \qquad \lambda' = \lambda * \min(N2,M)$
$g = p(0,N)+\ldots+p(0,1) + p(0,0) + p(1,N)$
$s = p\ (1,N)/N$
$q = p\ (1,N)$

We use the vector s to calculate later the utilization, and vector q to calculate the mean queue length.

4. Calculate the state probabilities of the second column, except of $p(1,0)$.

For $I=1,N-1$ $\qquad \lambda' = \lambda * \min\ (N2,M)$

$p(1,\ N-I) = \{p(0,N-I)(\lambda' + (N-I)\gamma+\alpha) - p(0,N-I-1)\alpha-$
$\quad -p(0,N+I-1)(N-I + 1)\gamma\}/\mu\beta(1,N-I)$

$s = s + p(1,N-I) * \min(1,N-I)/N$
$q = q + p(1,N-I)$
$g = g + p(1,N-I)$

5. Calculate $p(I,0)$, then the first state probability of the next column, and then the rest state probilities of that column.

For $I = 1,M-1$

$\lambda_2 = \lambda * \min(N2,M-I +1)$
$\lambda_1 = \lambda * \min(N2,M-I)$
$p(I,0) = \{p(I-1,0)\lambda_2 + p(I,1)\gamma\}/\lambda_1 + \alpha$
$g = g + p(I,0)$
$q = q + I * p(I,0)$
$p(I+1,N) = \{p(I,N)(\mu\beta(I,N) + \lambda_1 + N\gamma) - p(I-1,N)\lambda_2-$
$\quad -p(I,N-1)\alpha\}/\mu\beta(I+1,N)$
$g = g + p(I+1,N)$
$s = s+p\ (I+1,N) * \min(I+1,N)/N$
$q = q+(I+1)p(I+1,N)$

For $J = 1,N-1$

$p(I+1,N-J) = \{p(I,N-J)\{\lambda_1+\alpha+(N-J)\gamma+\mu\beta(I,N-J)\}-$
$\quad - p(I-1,N-J)\lambda_2 - p(I,N-J+1)\cdot(N-J+1)\gamma-$
$\quad - p(I,N-J-1)\alpha\}/\mu\beta(I+1,N-J)$
$g = g+p(I+1,N-J)$
$s = s+p(I+1,N-J) * \min(I+1,N-J)/N$
$q = q+(I+1)\ p(I+1,N-J)$

6. Calculate the state probability $p(M,0)$. (Last state on the last column.)

$$p(M,0) = \{p(M-1,0)\lambda+p(M,1)\gamma\}/\alpha$$
$$g = g + p(M,0)$$
$$q = q + MP(M,0)$$

7. Now for the last column we have:

$$d(N) = p(M,N)\{N\gamma+\mu\beta(M,N)\} - p(M-1,N)\lambda-p(M,N-1)\alpha$$

and for $I = 1,N-2$

$$d(N-I) = p(M,N-I)\{\alpha+ \mu\beta(M,N-I) + (N-I)\gamma\} - p(M-1,N-I)\lambda - p(M,N-I+1)(N-I+1)\gamma - p(M,N-I-1)\alpha$$

8. Finally we determine the state probabilities $P(0,N), \ldots, P(0,1)$ Hence we solve the system of N equations with N unknowns.

$$g^T\cdot\underline{P} = 1$$
$$d^T(N)\cdot\underline{P} = 0$$
$$\cdots\cdots\cdots\cdots$$
$$d^T(2)\cdot\underline{P} = 0$$

where $\underline{P} = \{P(0,N), P(0,N-1),\ldots,P(0,1)\}^T$

when we find $\underline{P}$, we can find the real utilization and mean queue length

$$\text{UTILIZATION} = S^T \cdot \underline{P}$$
$$\text{MEAN QUEUE LENGTH} = q^T\cdot\underline{P}$$
$$\text{THROUGHPUT} = \text{UTILIZATION} * \mu * N$$

END.

Using this algorithm we wrote a general FORTRAN program, which because of limited space cannot be presented here. In the next section we present our results for different cases.

IV. RESULTS

In the beginning we compare the throughput of a single-CPU system with the throughput of a double-CPU system (Fig. 5-Fig. 10).

Parameter μ will represent the service rate of the single CPU system, therefore each CPU in the double-CPU system has rate $\mu/2$.

All systems have 5 I/O devices. Each I/O device has a service rate $\lambda = 300$.

In all figures the vertical axis represents the ratio RAT ,

$$\text{RAT} = \frac{\text{Throughput of single-CPU sys.}}{\text{Throughput of the N-CPU sys.}}$$

while the horizontal axis shows the amount of availability (AV) in the system, as it was defined in section II.

For the first two cases (Fig. 5 and 6) we have the same number of programs, (five), but different service rates for the CPUs.

Fig. 5. Here $\mu = 300$. Our CPU is slower than our I/O devices. (Each I/O is as fast as the single-CPU, and faster than each CPU in the double-CPU system.). We can see that the Sauer and Chandy approximation is very good. Also for both $\alpha = 5$ and $\alpha = 2{,}000$ there is no significant difference in the performance of the system.

Fig. 6. Now $\mu = 600$. The CPU is faster than before. The approximate solution is again very good, although now we can see some difference in the performance as we change α.

In figures 7-10 we keep the CPU service rate the same in all cases, but we change the maximum number of programs in the system ($\mu = 1500$).

Fig. 7. Here we have 15 programs. Now the CPU is very fast but we have also a large number of programs. As in Fig. 6 the Sauer and Chandy approximation gives almost the same results with our analytical method.

Fig. 8,9,10. Now we start decreasing the max. number of programs in our system (12, 8 and 5). As we decrease the number of programs we see that the results we got with the approximation method differ more and more from our results. Now both values of 'α' and 'γ' play a significant role in the performance of our system.

As we increase 'α' there is also an increase in the throughput of both the single and double-CPU systems, but with different growth ratios. So after a certain value of the availability the quantity RAT for a system with small 'α' is smaller than the quantity RAT, for a system with large 'α'.

Hence a multi-processor system with small 'α' is preferable than a system with large 'α'. One should expect that, because for a constant value of the availability any time we increase 'α' we increase also 'γ', but while we can fix only one CPU at a time, more than one CPU may fail at any time.

Finally in figures 11 and 12 we compare the throughput of a single-CPU system with the throughput of a 4-CPU system. Again as before $\lambda = 300$.

In Fig. 11 $\mu = 1200$ and in Fig. 12 $\mu = 3000$. We see again similar results as before. As the CPU is getting faster the Sauer and Chandy results differ more from our analytical results.

V. SUMMARY AND CONCLUSIONS

In this paper we described an analytical model for an unreliable multi-miniprocessor system with an arbitrary number of CPUs and I/Os.

Using recursive techniques we were able to solve analytically for performance metrics of this system.

Comparing the performance of a single-CPU system with that of a double-CPU and a 4-CPU system, we found that in most cases the single-CPU system is better. (Fact related mostly to our constraint to be able to fix only one CPU at a time.)

The Chandy and Sauer method was very good when we had a large mean-CPU queue length (in relation to the I/O mean queue length), otherwise their results were very poor. But we can still use their results as a lower bound.

We cannot ignore the fact that there are two parameters ('α' and 'γ') that control the performance of an unreliable multi-miniprocessor system, especially when the availability of the system is large.

ACKNOWLEDGMENT

The author gratefully acknowledges the encouragement and support of Professor D. Towsley at the University of Massachusetts, Amherst, throughout the preparation of this work.

REFERENCES

1. Browne, J.C., Chandy, K.M., Hogarth, J. and Lee, C.C. "The Effect of Throughput of Multiprocessing in a Multiprogramming Environment" IEEE Transactions on Computers 22,8 (August 1973).

2. Herzog, V., Woo, L., and Chandy, K.M. "*Solution of Queueing Problems by a Recursive Technique*", IBM J.R.D. 19, 3 (May 1975).

3. Sauer, C.H., Chandy, K.M. "*Computer System Performance Modeling, A Primer*" Prentice-Hall.

4. Sauer, C.H., Chandy, K.M. "*Parametric Modeling of multi-miniprocessor Systems*", I.B.M. Research Report R.C. 5978 (#25935) 5/7/76 Computer Science.

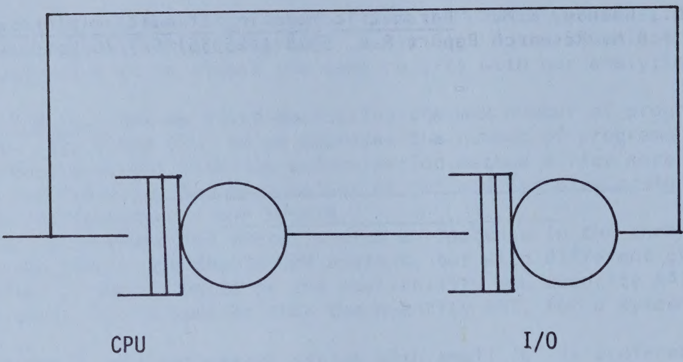


Fig. 1

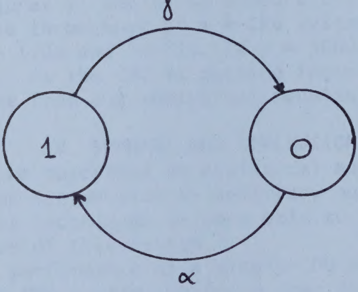


Fig. 2

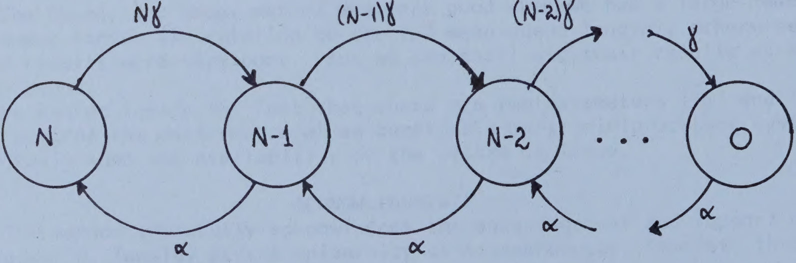
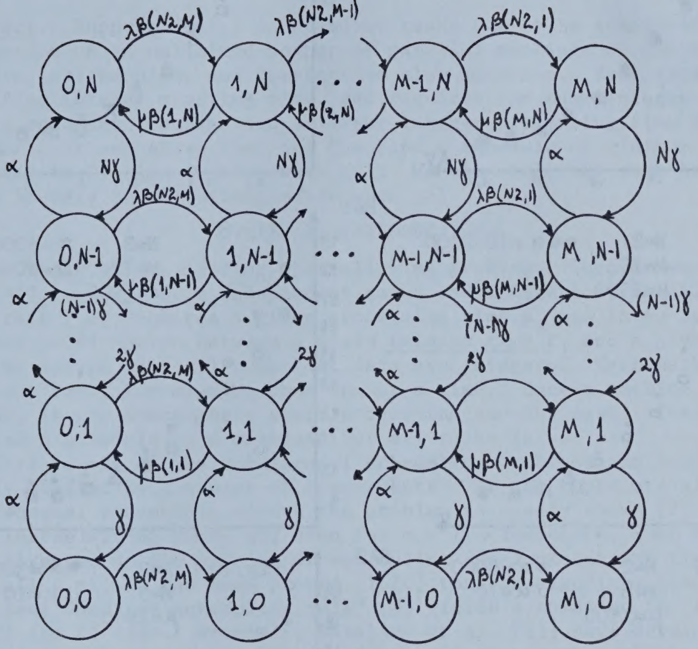

Nγ
(N-1)γ
(N-2)γ
γ
N
N-1
N-2
. . .
0
α
α
α
α

Fig.3


0,N
1,N
M-1,N
M,N
0,N-1
1,N-1
M-1,N-1
M ,N-1
0,1
1,1
M-1,1
M ,1
0,0
1,0
M-1,0
M , 0
λβ(N2,M)
λβ(N2,M-1)
λβ(N2,1)
μβ(1,N)
μβ(2,N)
μβ(M,N)
μβ(1,N-1)
μβ(M,N-1)
μβ(1,1)
μβ(M,1)
α
Nγ
(N-1)γ
2γ
γ

State : (I,J),I:Number of programs in the CPUs.
J:Number of working CPUs.
$\beta(m,n)$=MIN(m,n)

M:Max. number of programs in the system.
N:Number of CPUs
N2:Number of I/Os
λ:Rate of service in each I/O.
μ:Rate of service in each CPU.
γ:Rate of failures.
α:Rate of repairs.

Fig. 4

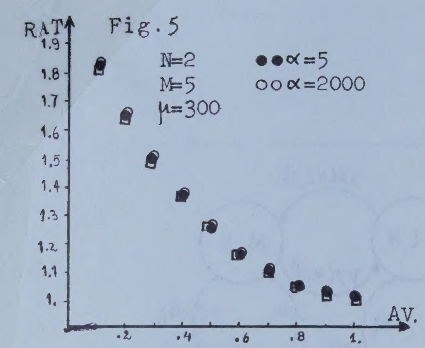


Fig.5

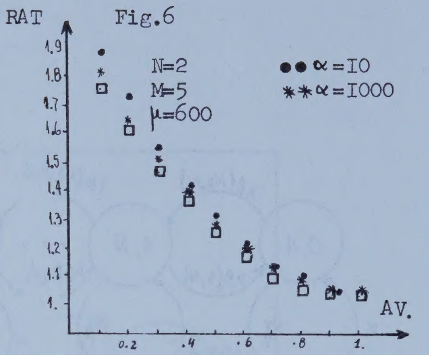


Fig.6

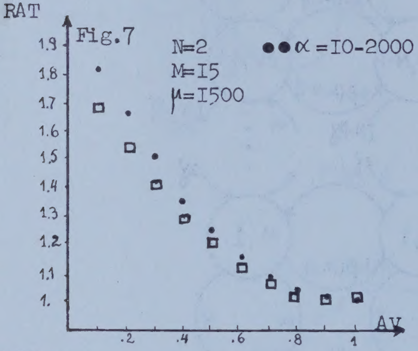


Fig.7

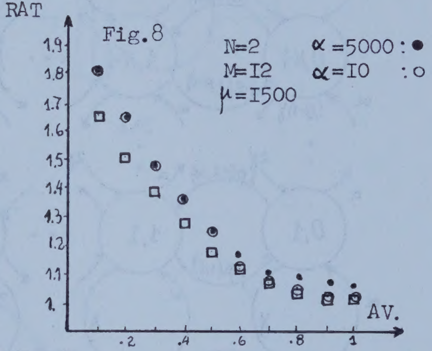


Fig.8

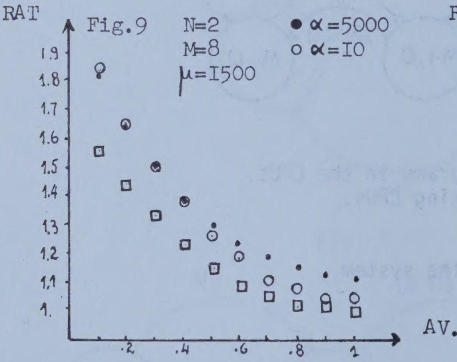


Fig.9

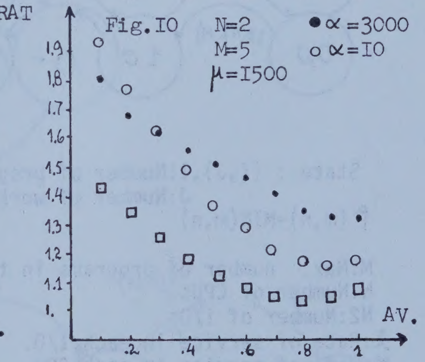


Fig.I0

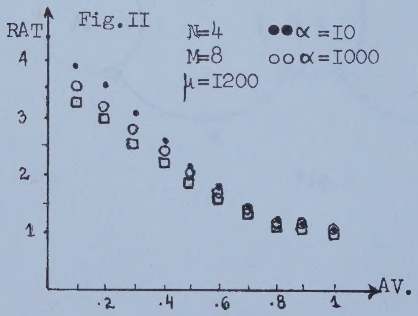


Fig.II

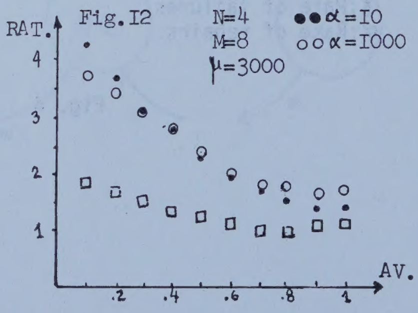


Fig.I2

□□□□: Sauer and Chandy method.

ON A SCHEDULING PROBLEM WITH TWO TYPES OF MACHINES*

KAZUO NAKAJIMA
EE/CS Dept.
Texas Tech Univ.
Lubbock, Texas 79409

S. LOUIS HAKIMI
EE/CS Dept.
Northwestern Univ.
Evanston, IL 60201

Abstract: Suppose that t independent tasks are to be scheduled without preemption on an unlimited number of parallel machines of two types: expensive fast machines and inexpensive slow machines. Each task has a specified integer starting time, and requires the same integer processing time q on a fast machine and a larger integer processing time on a slow machine. It was shown that one can find a schedule of minimum total machine cost in O(t log t) time when q=1. We show that the same problem becomes NP-hard in the strong sense when q>1.

1. INTRODUCTION

We begin by considering the following problem. Suppose we are given a set $\mathcal{T}=\{T_1,\ldots,T_t\}$ of t independent tasks and m identical parallel machines. Each task $T_i \varepsilon T$ requires a given processing time p_i and is to be processed without interruption between a given release time r_i and a given deadline d_i. We assume that all numerical data are integers. Each machine can process any task, but no more than one at a time. Once a machine has completed a task, it can immediately start processing another task. The problem is to find a schedule on m machines for all tasks in $\mathcal{T}$.

This is one of the fundamental scheduling problems and has been extensively studied by a number of researchers. If the tasks are allowed to have unequal processing times, the problem is easily shown [2] to be NP-hard in the strong sense [3] even for m = 1. Therefore, much attention has been directed at the problem of scheduling equal processing time tasks. When all $p_i=1$, it has been proved [1,10] that an algorithm based on the "earliest deadline scheduling rule" [8] yields a solution for the problem in O(t log t) time. Recently, Nakajima et al. [11] have developed a different O(t log t) algorithm which finds the minimum number of machines needed to process all unit time tasks in $\mathcal{T}$ as well as a corresponding schedule of the tasks. When all $p_i=p>1$, Simons [13] first presented an $O(t^2 \log t)$ algorithm for m = 1. Garey et al. [4] have recently obtained an O(t log t) algorithm for the same case. Furthermore, Simons [14] has more recently shown that the problem is solvable in time $O(t^3 \log t)$ for the general m-machine case.

On the other hand, Gertsback et al. [5] considered another interesting special case of the problem in which all $r_i=d_i-p_i$. For this case several O(t log t) algorithms for finding a schedule on the least number of machines have been reported [6,7,9]. Recently, Nakajima et al. [11] considered a natural generalization of this problem by allowing two types of machines--slow machines each with cost C^s and fast machines each with cost $C^f>C^s$--which are capable of processing any task in $\mathcal{T}$. Each task $T_i \varepsilon \mathcal{T}$ has a specified starting time $s_i=r_i=d_i-p_i$ and requires a processing time p_i on a slow machine or a processing time $q_i<p_i$ on a fast machine. Then the problem is to find a schedule of minimum total machine cost to process all tasks in $\mathcal{T}$. It was proved [11] that the problem is NP-hard in the strong sense even when all $p_i/q_i=C^f/C^s$. However, a special case in which all $q_i=1$ and all p_i are arbitrary, was shown to be solvable in O(t log t) time.

As in the case of identical machines, one may consider the above special case when all $q_i=q>1$. In this paper, we prove that in contrast with the previous case, this special case becomes NP-hard in the strong sense.

---

* This work was supported by NSF Grant ENG79-09724.

2. NP-HARDNESS OF THE PROBLEM

It is assumed in our problem that once a machine has completed a task, it can immediately start another task. Therefore, each task $T_i \varepsilon \mathcal{T}$ can be represented by a pair of semi-open intervals called an f-interval and an s-interval. In other words, $T_i=\langle F(T_i),S(T_i)\rangle$, where $F(T_i)=[s_i,s_i+q_i)$ and $S(T_i)=[s_i,s_i+p_i)$. Let n^f and n^s be the numbers of fast machines and slow machines used to process all tasks in $\mathcal{T}$. Then we will say that there is an (n^f,n^s) assignment for $\mathcal{T}$. The cost of this assignment is $n^fC^f+n^sC^s$. Let C* be a designated total cost. Then the "Fixed Starting Time" scheduling problem is stated as follows:

> FST: Given $\mathcal{T}$, C^f, C^s and C*, are there integers n^f
> and n^s such that there is an (n^f,n^s) assignment
> for $\mathcal{T}$ with $n^fC^f+n^sC^s \leq C^*$?

In this section we will prove that the FST problem is NP-hard in the strong sense when all $q_i=q>1$. The proof is based on a transformation from the following NP-complete ONE-IN-THREE 3SAT (in short, 1-3SAT) problem [12] to the FST problem.

> 1-3SAT: Given a set U of Boolean variables and a collection C of
> clauses over U with each clause in C containing exactly three
> literals, is there a truth assignment to the variables
> in U such that each clause has exactly one true literal?

We begin our discussion by considering an arbitrary instance $\langle U,C\rangle$ of the 1-3SAT problem. Then, we construct a task system $\mathcal{T}(U,C)$ to correspond to the instance $\langle U,C\rangle$ of the 1-3SAT problem. The tasks in $\mathcal{T}(U,C)$ are to be processed by a collection of fast machines and slow machines. We will then prove that all tasks in $\mathcal{T}(U,C)$ can be assigned to the fast machines and slow machines with the total cost not exceeding a given cost C*, if and only if, $\langle U,C\rangle$ has a solution.

The construction of $\mathcal{T}(U,C)$: Let $U=\{u_1,\ldots,u_n\}$ and $C=\{c_1,\ldots,c_m\}$ and denote the set of leterals $\{u_1,\ldots,u_n\}\cup\{\overline{u}_1,\ldots,\overline{u}_n\}$ by $U \cup \overline{U}$. Our task system $\mathcal{T}(U,C)$ will consist of a large number t of tasks. The fast machine processing time Π of all tasks in $\mathcal{T}(U,C)$ will be equal to 10n; thus the length of the f-interval of any task in $\mathcal{T}(U,C)$ is equal to 10n. This enables us to describe the task system $\mathcal{T}(U,C)$ by giving the s-interval for each of the t tasks.

Almost all f-intervals in $\mathcal{T}(U,C)$ are given in forms of "blocks." The structure of a block is shown in Fig. 1. Observe that the block in Fig. 1 consists of 2n identical rectangles each of height 1 and width w. These rectangles are stacked on top of each other in a staggered fashion forming "stairs" on each side of the block. In a basic block, each rectangle represents a task whose f-interval matches with a horizontal line of that rectangle. (In an abuse of the terminology, the term "block" is used as the block itself or as the set of tasks in that block.) Thus in a basic block, we have exactly 2n tasks all having processing time w=10n by a fast machine. Each task in a basic block has a starting time one unit later than that of the task corresponding to the rectangle immediately above it.

Our entire task system $\mathcal{T}(U,C)$ consists of two collections of tasks: (1) the block tasks, and (2) the auxiliary tasks.

(1) The collection of the block tasks consists of 5m+14 cascaded blocks jammed against each other as depicted in Fig. 2. These blocks are denoted by H_1 through H_9, and A_i,B_i,D_i,E_i and $G_i(i=1,..,m+1)$. We first observe that there are 2n rectangles (not necessarily 2n tasks in this case) in each of the blocks. Consider the top rectangles in each of the blocks. Note that these rectangles together form a horizontal "strip" across the top of the cascaded block structure. We denote this top strip by L_1 and the strip formed by the rectangles immediately below it by $L_2,\ldots$, and the strip formed by the bottom rectangles in the cascaded block structure by L_{2n}. For each integer i, we associate, respectively, strips L_{2i-1} and

L_{2i} with the variable u_i and its complement $\overline{u}_i (i=1,\dots,n)$. Thus, to each literal $u \varepsilon U \cup \overline{U}$ there is associated a unique strip, to be denoted by $L(u)$, in the cascaded block structure.

All blocks but H_5 are basic blocks. Thus, these blocks give us the f-intervals of $2n(5m+13)$ tasks in $T(U,C)$. For a basic block A_j $(j=1,\dots,m)$, we denote the task on strip $L(u)$ in this block by $a_j(u)$ for each $u \varepsilon U \cup \overline{U}$. Similarly, we define $b_j(u)$, $d_j(u)$, $e_j(u)$, $g_j(u)$ $(j=1,\dots,m)$, and $h_i(u)$ $(i=1,\dots,4,6,\dots,9)$.

Block H_5 is different from the basic blocks. It is of width $10n+1$ and has exactly n tasks, instead of 2n, whose f-intervals are all exactly 10n long. The f-intervals of these n tasks, denoted by $h_5(u_i,\overline{u}_i) (i=1,\dots,n)$, in the block H_5 are shown in Fig. 3 by the shaded rectangles. More precisely, the f-interval of the task $h_5(u_i,\overline{u}_i)$, denoted by $F(h_5(u_i,\overline{u}_i))$ shown on strip L_{2i-1} in Fig. 3, is:

$$F(h_5(u_i,\overline{u}_i)) = [40n + 2i - 1,\ 50n + 2i - 1)(i = 1,\dots,n).$$

To describe the s-intervals of the block tasks (as well as the f- and s-intervals of the auxiliary tasks) we must use explicitly the instance $\langle U,C \rangle$ of the 1-3SAT problem. We note that the left endpoint of the s-interval of each task is identical to that of the horizontal line of its corresponding rectangle. Thus, we will give only the length of each s-interval.

The length of the s-interval of each task in blocks H_1 and H_5-H_9 is 30n. That of each task in H_2-H_4 is $40n+1$.

The blocks A_j, B_j, D_j, E_j and G_j correspond to clause $c_j \varepsilon C$ of $\langle U,C \rangle$ $(j=1,\dots,m)$. Let the literals in the clause c_j be x_j, y_j and $z_j \ \varepsilon\ U \cup \overline{U}$ which are by assumption distinct. For each $u \varepsilon U \cup \overline{U} - \{x_j,y_j,z_j\}$ the s-intervals of the tasks $a_j(u), b_j(u), d_j(u), e_j(u)$ and $g_j(u)$, which are by definition on strip $L(u)$ in the blocks A_j, B_j, D_j, E_j and G_j, are all of length 30n. For each $u \varepsilon \{x_j,y_j,z_j\}$ the s-intervals of the tasks $a_j(u)$ and $b_j(u)$ are 40n long each and those of the tasks $d_j(u), e_j(u)$ and $g_j(u)$ are, respectively, 23n, 17n and 30n long. Some of these s-intervals are shown in Fig. 4 by the broken lines with arrows. The s-intervals of the tasks in blocks $A_{m+1}, B_{m+1}, D_{m+1}, E_{m+1}$ and G_{m+1} are all of length 30n. The above description and the examinations of Figs. 1-4 provide the necessary information for $n(10m+27)$ block tasks in $T(U,C)$.

(2) The collection of the auxiliary tasks consists of m+1 sets $R_0, R_1, \dots, R_m$. Let $S(T)$ denote the s-interval of task T. To begin with, R_0 consists of 18 tasks r_0^{11}-r_0^{16}, r_0^{21}-r_0^{26} and r_0^{31}-r_0^{36} as indicated in Fig. 2, where in that figure (in contrast with the block tasks) the s-interval of each auxiliary task is represented by a rectangle and its f-interval appears as a front part of that rectangle whose end is indicated by a broken line. More precisely,

$$S(r_0^{k\ell}) = [20n\ell - 55n,\ 20n\ell - 35n) \text{ for } k = 1,2,3 \text{ and } \ell = 3,4,5,6,$$

$$S(r_1^{k1}) = [85n,\ 195n) \text{ for } k = 1,2,3,$$

$$S(r_0^{12}) = S(r_0^{22}) = [105n,\ 120n + 1), \text{ and}$$

$$S(r_0^{32}) = [105n,\ 133n + 1).$$

The auxiliary tasks in $R_j (j=1,\dots,m)$ are associated with blocks A_j, B_j, D_j, E_j and G_j and each R_j contains nine tasks r_j^{11}-r_j^{31}, r_j^{12}-r_j^{32}, $r_j(x_j), r_j(y_j)$ and $r_j(z_j)$ as shown in Fig. 4, where the representations of the s-interval and f-interval of each auxiliary task are as in Fig. 2. The s-intervals of the first six tasks in R_j are explicitly specified as follows:

$$\left.\begin{aligned} S(r_j^{k1}) &= [50nj + 89n,\ 50nj + 105n) \\ S(r_j^{k2}) &= [50nj + 105n,\ 50nj + 120n + 1) \end{aligned}\right\} \text{ for } k = 1,2,$$

$$S(r_j^{31}) = [50nj + 103n,\ 50nj + 117n), \text{ and}$$

$$S(r_j^{32}) = [50nj + 117n,\ 50nj + 133n + 1).$$

To describe the remaining three tasks $r_j(x_j), r_j(y_j)$ and $r_j(z_j)$ in R_j, we need some additional notation. Let $R[S(d_j(x_j))]$ be the right endpoint of the s-interval of the task $d_j(x_j)$. Similarly define $R[S(d_j(y_j))]$ and $R[S(d_j(z_j))]$. Then the s=intervals of the tasks $r_j(x_j), r_j(y_j)$ and $r_j(z_j)$ are given by

$$S(r_j(v_j)) = [R[S(d_j(v_j))],\ R[S(d_j(v_j))] + 17n), \text{ for } v_j \varepsilon \{x_j, y_j, z_j\}.$$

This completes the description of our task system $\mathcal{T}(U,C)$. The reader should note that there are exactly $t=n(10m+27)+9m+18$ tasks in $\mathcal{T}(U,C)$.

We have now constructed a task system $\mathcal{T}(U,C)$ corresponding to an arbitrary instance $\langle U,C\rangle$ of the 1-3SAT problem. This construction can obviously be carried out in polynomial time.

Let $C^*=nC^f+(3n+3)C^s$ and assume that $3C^s<C^f<4C^s$. The following theorem states that there is a machine assignment for $\mathcal{T}(U,C)$ within the cost C^* if and only if $\langle U,C\rangle$ has a solution. Furthermore, as it is easily verified that the largest number in $\mathcal{T}(U,C)$ is bounded by a polynomial in t, this in fact proves that the FST problem is NP-hard in the strong sense.

THEOREM 1. Let $\langle U,C\rangle$ and $\mathcal{T}(U,C)$ be as above. Then, there is a truth assignment for U such that each clause in C has exactly one true literal if and only if there is an (n^f,n^s) assignment for $\mathcal{T}(U,C)$ with $n^fC^f+n^sC^s\le C^*$.

Proof. We first assume that $\langle U,C\rangle$ has a solution and prove that there is an (n^f,n^s) assignment for $\mathcal{T}(U,C)$ with $n^fC^f+n^sC^s\le C^*$. Let $TA: U\cup\overline{U}\to\{\text{true}, \text{false}\}$ be a truth assignment function such that each clause in C has exactly one true literal. We now show that there is an $(n,3n+3)$ assignment for $\mathcal{T}(U,C)$ with the cost C^* as follows. If TA(u)=true with $u=u_i$ or $\overline{u}_i$, then all tasks on strip L(u) are assigned to the same fast machine N_i. Also N_i is assigned the i-th task $h_5(u_i,\overline{u}_i)$ in block H_5 which is on strip L_{2i-1}. We make the following observations: (1) N_i can process all tasks assigned to it, (2) there are n fast machines and either all tasks on strip $L(u_i)$ or all tasks on $L(\overline{u}_i)$ are assigned to a fast machine, and (3) all tasks in block H_5 are assigned to fast machines.

To describe the slow machines' task assignment, we assume that in every clause $c_j=(x_j+y_j+z_j)$ in C the literal x_j represents the true literal and the literals y_j and z_j represent the false literals. This implies that all tasks on strip $L(x_j)$ are assigned to a fast machine. We now proceed as follows. The tasks r_0^{1k} (resp., r_0^{2k})(k=2,3,4,6) and the tasks r_j^{11} and r_j^{12} (resp., r_j^{21} and r_j^{22})(j=0,1,...,m+1) as well as the tasks $e_j(y_j)$ (resp., $e_j(z_j)$)(j=1,...,m) are assigned to slow machine M_1'(resp., M_2'). The tasks r_0^{3k}(k=3,4,5,6) and the tasks r_j^{31} and r_j^{32}(j=0,1,...m+1) as well as the tasks $r_j(x_j)$(j=1,...,m) are assigned to slow machine M_3'. The reader can convince himself that these three slow machines can process the tasks assigned to them. The assignments for the remaining 3n slow machines are as follows. If TA(u)=false with $u=u_i$ or $\overline{u}_i$, then all the remaining tasks on L(u) are assigned to three slow machines M_{i1}, M_{i2} and M_{i3}. Furthermore, if the above false literal u is equal to y_j(resp., z_j)(j=1,...,m), then the one of the slow machines M_{i1}, M_{i2} and M_{i3}, in particular, the one that processes the task $d_j(y_j)$(resp., $d_j(z_j)$), also processes the task $r_j(y_j)$(resp.,

$r_j(z_j))(j=1,\ldots,m)$.

The proof of the validity of the above (n,3n+3) assignment is tedious but straightforward and thus is left out. We have now shown that if $\langle U,C\rangle$ has a solution, then there is a machine assignment for $T(U,C)$ within the cost C*.

We need to prove that if there is an (n^f,n^s) assignment for $T(U,C)$ with $n^f C^f + n^s C^s \leq C^*$, then $\langle U,C\rangle$ has a solution. This is more difficult and requires a number of lemmas, which are given below without proofs.

LEMMA 1. Let an (n^f,n^s) assignment be a least cost assignment of the tasks in $T(U,C)$. If $n^f C^f + n^s C^s \leq C^*$, then $n^f=n$ and $n^s=3n+3$.

Proof. Consider the set T_0 containing the tasks in blocks H_2-H_5 and the three auxiliary tasks in $R_0^4=\{r_0^{14}, r_0^{24}, r_0^{34}\}$. We will demonstrate that even a least cost machine assignment of the tasks in T_0 requires n fast and 3n+3 slow machines. This would prove that any deviation from an (n,3n+3) assignment would increase the cost beyond C* for T_0 and, of course, for $T(U,C)$.

Suppose n^f fast machines are available and that $\ell \leq 3$ of them are processing ℓ tasks in R_0^4. Since the f-intervals of all tasks in $H_3 \cup R_0^4$ overlap with each other, at most $n^f-\ell$ tasks in H_3 can be assigned to fast machines. Similarly, at most $n^f-\ell$ tasks in H_4 can be assigned to fast machines. It is also clear that these n^f fast machine can process at most n^f tasks in H_2 and at most $\min(n,n^f)$ tasks in H_5. Then we are left with, at least, $2n-(n^f-\ell)$ tasks in H_3, $2n-(n^f-\ell)$ tasks in H_4, $n-\min(n,n^f)$ tasks in H_5, $2n-n^f$ tasks in H_2, and $3-\ell$ tasks in R_0^4. As the s-intervals of all these tasks overlap with each other, the number n^s of slow machines must satisfy

$$n^s \geq 7n + \ell + 3 - 3n^f - \min(n,n^f). \qquad (1)$$

Using the hypothesis and (1), we obtain

$$C^* = nC^f + (3n+3)C^s \geq n^f C^f + n^s C^s$$

$$\geq n^f C^f + (7n + \ell + 3 - 3n^f - \min(n,n^f))C^s,$$

or

$$(n - n^f)C^f \geq (4n + \ell - 3n^f - \min(n,n^f))C^s. \qquad (2)$$

Suppose $n^f > n$. Then (2) is written as

$$(n - n^f)C^f \geq (3n + \ell - 3n^f)C^s.$$

Since $\ell \geq 0$, we obtain

$$(n - n^f)C^f \geq (3n - 3n^f)C^s,$$

or

$$C^f \leq 3C^s.$$

This, however, violates the intercost relationship, $3C^s < C^f < 4C^s$. On the other hand, suppose $n_f < n$. Then, from (2), we obtain

$$(n - n_f)C^f \geq (4n + \ell - 4n_f)C^s$$

$$\geq (4n - 4n_f)C^s,$$

or $C^f \geq 4C^s$, which again violates the intercost relationship. Thus, $n_f=n$. Then, (1) implies that $n_s \geq 3n + \ell + 3$; but as $n^f C^f + n^s C^s \leq C^*$, $\ell = 0$ and $n^s = 3n + 3$.

From the proof of Lemma 1 we observe that to achieve an (n,3n+3) assignment for T_0, one must indeed assign exactly n tasks from each of blocks H_2-H_5 to n fast machines and exactly n tasks from each of blocks H_2-H_4 to 3n slow machines and the three tasks r_0^{14}, r_0^{24} and r_0^{34} also to 3 slow machines. We denote the last three slow machines by M_1', M_2' and M_3'.

LEMMA 2. If there is an (n,3n+3) assignment for $T(U,C)$, then all tasks in R_0 are assigned to the three slow machines M_1', M_2' and M_3'.

Lemma 2 implies that if there is an (n,3n+3) assignment for TU,C), then there is an (n,3n) assignment of all the block tasks in H_1-H_9, A_1, B_1 and D_1.

LEMMA 3. If there is an (n,3n) assignment of the tasks in H_1-H_9, then, for i=1,...,2n, either (a) four tasks on strip L_i in blocks H_6-H_9 (resp., H_1-H_4) are assigned to the same fast machine, or (b) two of the four tasks on strip L_i, namely those in H_6 and H_9 (resp., H_1 and H_4) are assigned to the same slow machine and the other two in H_7 and H_8 (resp., H_2 and H_3) are assigned to two other slow machines.

As a consequence of the proof of Lemma 1, we know that each task in block H_5 must be processed by a unique fast machine. Let N_i be the fast machine which processes the i-th task $h_5(u_i,\overline{u}_i)$ from the top, that is, the one on strip L_{2i-1}, in block H_5(i=1,...,n).

LEMMA 4. If there is an (n,3n) assignment of the tasks in blocks H_1-H_9, then, for i=1,...,n, either all tasks on strip $L(u_i)$ in blocks H_6-H_9 are assigned to N_i, or all tasks on strip $L(\overline{u}_i)$ in blocks H_6-H_9 are assigned to N_i. Furthermore, not both the tasks on $L(u_i)$ and there on $L(\overline{u}_i)$ in blocks H_6-H_9 can be assigned to fast machines.

LEMMA 5. Consider the set $\mathcal{T}_j$ of the tasks in blocks G_j, A_{j+1}, B_{j+1}, D_{j+1} and E_{j+1} and the nine auxiliary tasks in R_j(j=1,...,m). If there is an (n,3n+3) assignment for $\mathcal{T}_j$, then these auxiliary tasks are all assigned to slow machines. Furthermore, three slow machines are required to process the six auxiliary tasks in $\{r_j^{i\ell} | i=1,2,3, \text{ and } \ell=1,2\}$.

Let $\mathcal{T}'_j$ be the set containing all tasks in blocks G_{j-1}, A_j, B_j, D_j, Ej, G_j, A_{j+1}, B_{j+1}, D_{j+1} and the nine auxiliary tasks in R_j and the six auxiliary tasks in $\{r_{j-1}^{i\ell} | i=1,2,3, \text{ and } \ell=1,2\}$(j=1,...,m), where $G_0 = H_9$.

LEMMA 6. Suppose that there is an (n,3n+3) assignment for $\mathcal{T}'_j$. Then we have: (a) If any task on some strip L_i in $\mathcal{T}'_j$ is assigned to a fast machine, then all tasks on L_i in $\mathcal{T}'_j$ are assigned to the same fast machine (i=1,..., 2n); and (b) The tasks in one of the three sets $\{d_j(x_j),e_j(x_j)\}$, $\{d_j(y_j), e_j(y_j)\}$ and $\{d_j(z_j),e_j(z_j)\}$ are assigned to a fast machine and those in the other two sets are assigned to slow machines.

We are now prepared to prove the remaining part of Theorem 1 that if there is an (n^f,n^s) assignment for $\mathcal{T}(U,C)$ with $n^fC^f+n^sC^s \leq C^*$, then there is a truth assignment for U such that each clause in C has exactly one true literal.

Proof of Theorem 1 (continued). By Lemma 1, an (n,3n+3) assignment for $\mathcal{T}(U,C)$ is the only assignment for which the cost $\leq C^*$. By Lemma 2, we have an (n,3n) assignment for the tasks in blocks H_1-H_9. Then, by Lemma 4, for each pair of strips $L(u_i)$ and $L(\overline{u}_i)$(i=1,...,n), either all tasks on strip $L(u_i)$ or all tasks on $L(\overline{u}_i)$ in blocks H_6-H_9 are assigned to the same fast machine, but not both of them are assigned to fast machines. Thus, if we assign the true value to each literal $u \in U \cup \overline{U}$ such that all tasks on strip L(u) in blocks H_6-H_9 are assigned to a fast machine, then we will have a consistent truth assignment. By the successive application of Part (a) of Lemma 6, we see that if the tasks in blocks H_6-H_9 on some strip L_i are assigned to a fast machine, then all tasks on L_i in $\mathcal{T}(U,C)$ beyond H_9 are assigned to the same fast machine. Then by Part (b) of Lemma 6, we see that exactly one literal from each clause is assigned the true value. This completes the proof of the theorem.

3. CONCLUSIONS

In this paper we have considered the FST problem for the case in which all tasks have non-unit equal processing times on a fast machine. In contrast with the result for the case where all tasks have unit processing times on a fast machine [12], we have shown that for the case at hand the problem is NP-hard in the strong sense, using a transformation form the 1-3SAT problem. The reader can easily check that there are s-intervals of thirteen different lengths in the task system constructed in the last section. Thus, we have in fact shown a stronger result in the sense that the processing times on a slow machine are drawn from a set of integers with fixed cardinality.. We believe that we could reduce such a cardinality from 13 to five if we modified our task system properly. However, we conjecture a much stronger result that for the case considered here the problem is NP-hard in the strong sense even when the tasks have two possible processing times on a slow machine. Furthermore, it was proved in [12] that the FST problem for the case where the ratio of the processing times on a slow machine versus a fast machine is fixed for all tasks is NP-hard in the strong sense, provided the fast machine processing times are arbitrary. Thus, there arises an interesting problem, that is, whether the FST problem is NP-hard in the strong sense or solvable in polynomial time when all tasks have the same non-unit processing time on a fast machine, the same larger processing time on a slow machine.

4. REFERENCES

1. J. Blazewicz, "Simple Algorithms for Multiprocessor Scheduling to Meet Deadlines," Info. Processing Letters, Vol. 6, pp. 162-164.
2. M. R. Garey and D. S. Johnson, "Two-Processor Scheduling with Start-Times and Deadlines," SIAM J. Computing, Vol. 6, pp. 414-426, 1977.
3. M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, San Francisco, CA, 1979.
4. M. R. Garey, D. S. Johnson, B. B. Simons, and R. E. Tarjan, "Scheduling Unit-Time Tasks with Arbitrary Release Times and Deadlines," SIAM J. Computing, Vol. 10, pp. 256-269, 1981.
5. I. Gertsbakh and H. I. Stern, "Minimal Resources for Fixed and Variable Job Schedules," Operations Res., Vol. 26, pp. 68-85, 1978.
6. U. I. Gupta, D. T. Lee, and J. Y.-T. Leung, "An Optimal Solution for Channel-Assignment Problem," IEEE Trans. Computers, Vol. C-28, pp. 807-810, 1979.
7. A. Hashimoto and J. E. Stevens, "Wire Routing by Optimizing Channel Assignment within Large Apertures," Proc. 8th Design Automation Workshop, 1971, pp. 155-169.
8. J. R. Jackson, "Scheduling a Production Line to Minimize Maximum Tardiness," Research Report 43, Management Science Research Project, Univ. of California, Los Angeles, CA, 1955.
9. B. W. Kernighan, D. G. Schweikert, and G. Persky, "An Optimum Channel-Routing Algorithm for Polycell Layouts of Integrated Circuits," Proc. 10th Design Automation Workshop, Portland, OR, June 1973, pp. 50-59.
10. T. Lang and E. B. Fernandez, "Scheduling of Unit-Length Independent Tasks with Execution Constraints," Info. Processing Letters, Vol. 4, pp. 95-98, 1976.
11. K. Nakajima, S. L. Hakimi, and J. K. Lenstra, "Complexity Results for Scheduling Tasks in Fixed Intervals on Two Types of Machines," SIAM J. Computing, to appear.
12. T. J. Schaefer, "The Complexity of Satisfiability Problems," Proc. 10th Ann. ACM Symp. on Theory of Computing, San Diego, CA, May 1978, pp. 216-226.
13. B. Simons, "A Fast Algorithm for Single Processor Scheduling," Proc. IEEE 19th Ann. Symp. on Foundations of Computer Science, Ann Arbor, MI, Oct. 1978, pp. 246-252.
14. B. Simons, "A Fast Algorithm for Multiprocessor Scheduling," Proc. IEEE 21st Ann. Symp. on Foundations of Computer Science, Long Beach, CA, Oct. 1980, pp. 50-53.

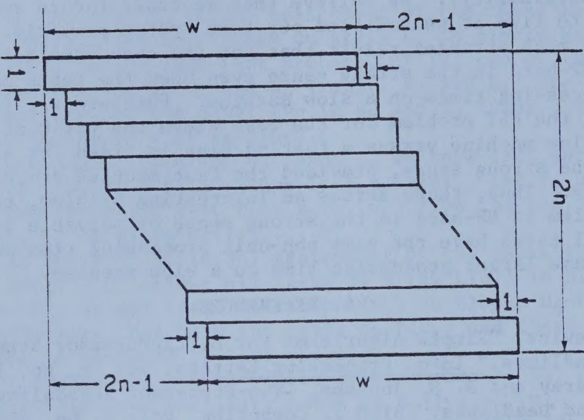

w
2n-1
1
1
1
2n
1
1
2n-1
w

Fig. 1. A block of height 2n and width w.

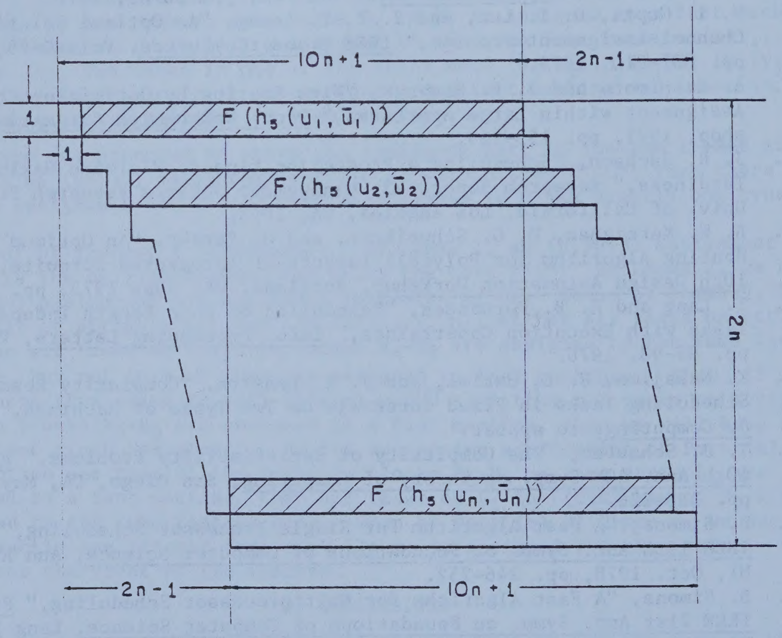

10n+1
2n-1
1
F (h₅ (u₁, ū₁))
1
F (h₅ (u₂, ū₂))
2n
F (h₅ (uₙ, ūₙ))
2n-1
10n+1

Fig. 3. Block H_5.

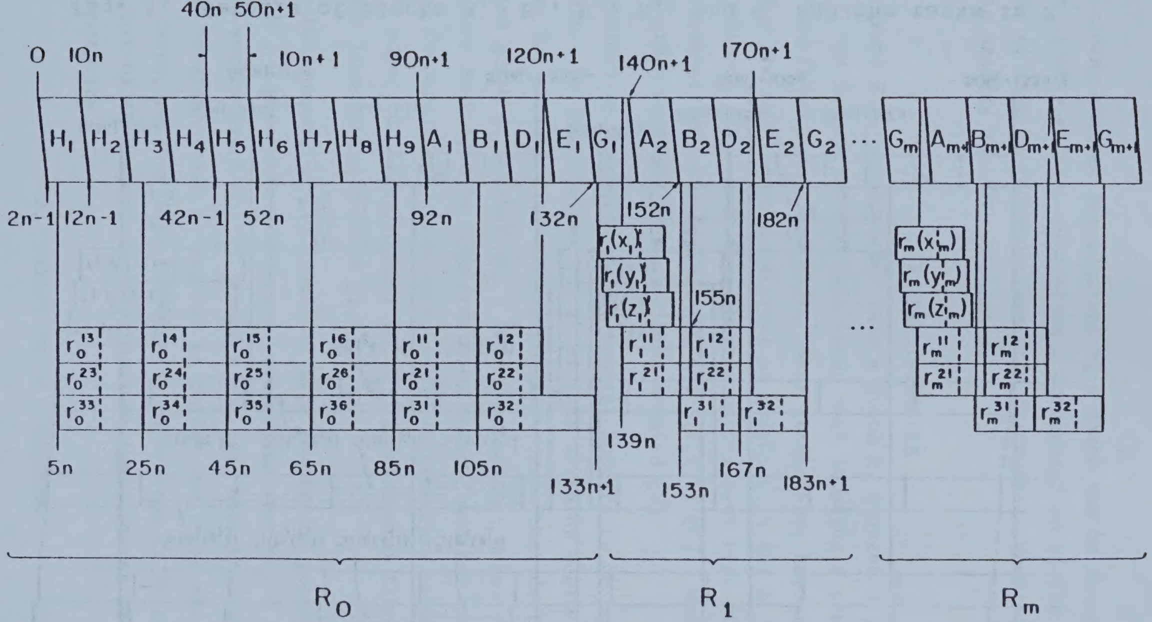

0
10n
40n
50n+1
10n+1
90n+1
120n+1
140n+1
170n+1
H₁ H₂ H₃ H₄ H₅ H₆ H₇ H₈ H₉ A₁ B₁ D₁ E₁ G₁ A₂ B₂ D₂ E₂ G₂ … Gₘ Aₘ₊₁ Bₘ₊₁ Dₘ₊₁ Eₘ₊₁ Gₘ₊₁
2n-1
12n-1
42n-1
52n
92n
132n
152n
182n
155n
r₁(x₁)
r₁(y₁)
r₁(z₁)
rₘ(x'ₘ)
rₘ(y'ₘ)
rₘ(z'ₘ)
r₀¹³ r₀¹⁴ r₀¹⁵ r₀¹⁶ r₀¹¹ r₀¹²
r₀²³ r₀²⁴ r₀²⁵ r₀²⁶ r₀²¹ r₀²²
r₀³³ r₀³⁴ r₀³⁵ r₀³⁶ r₀³¹ r₀³²
r₁¹¹ r₁¹² r₁²¹ r₁²² r₁³¹ r₁³²
rₘ¹¹ rₘ¹² rₘ²¹ rₘ²² rₘ³¹ rₘ³²
5n
25n
45n
65n
85n
105n
133n+1
139n
153n
167n
183n+1
R₀
R₁
Rₘ

Fig. 2. The task systemT(U,C).

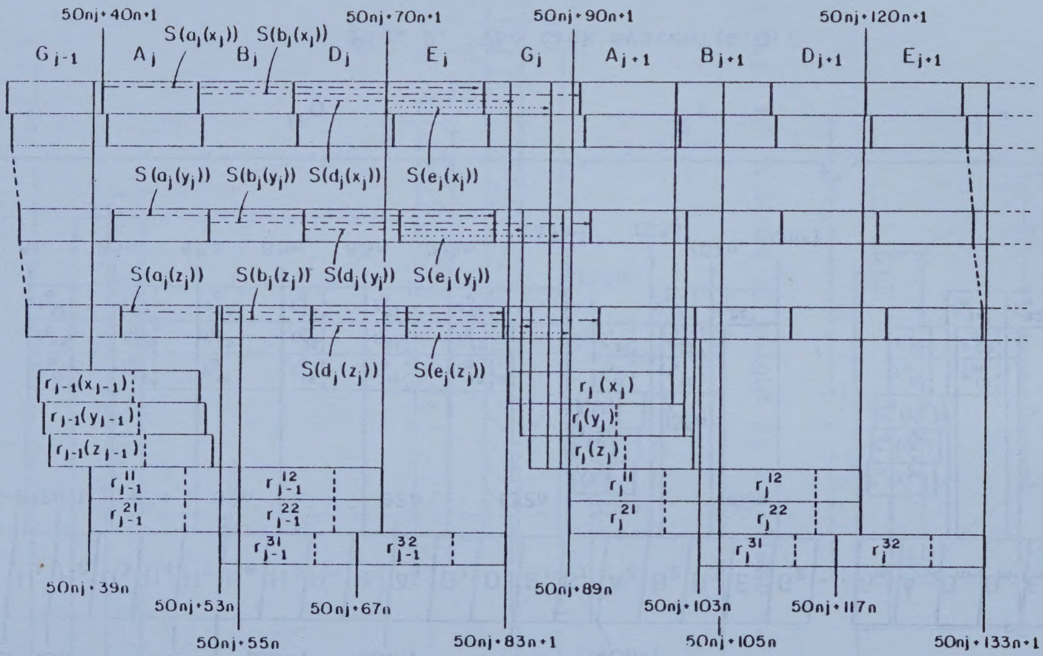

50nj+40n+1
50nj+70n+1
50nj+90n+1
50nj+120n+1
G_{j-1}
A_j
$S(a_j(x_j))$
$S(b_j(x_j))$
B_j
D_j
E_j
G_j
A_{j+1}
B_{j+1}
D_{j+1}
E_{j+1}
$S(a_j(y_j))$
$S(b_j(y_j))$
$S(d_j(x_j))$
$S(e_j(x_j))$
$S(a_j(z_j))$
$S(b_j(z_j))$
$S(d_j(y_j))$
$S(e_j(y_j))$
$S(d_j(z_j))$
$S(e_j(z_j))$
$r_{j-1}(x_{j-1})$
$r_{j-1}(y_{j-1})$
$r_{j-1}(z_{j-1})$
$r_j(x_j)$
$r_j(y_j)$
$r_j(z_j)$
r^{11}_{j-1}
r^{12}_{j-1}
r^{21}_{j-1}
r^{22}_{j-1}
r^{31}_{j-1}
r^{32}_{j-1}
r^{11}_j
r^{12}_j
r^{21}_j
r^{22}_j
r^{31}_j
r^{32}_j
50nj+39n
50nj+53n
50nj+55n
50nj+67n
50nj+83n+1
50nj+89n
50nj+103n
50nj+105n
50nj+117n
50nj+133n+1

Fig. 4. Details of Blocks A_j, B_j, D_j, E_j, and G_j and the tasks in R_j.

SOME NP-COMPLETE PROBLEMS ON GRAPH DECOMPOSITIONS

CHARLES J. COLBOURN
University of Saskatchewan, Saskatoon, Canada.

ABSTRACT

Deciding whether the edge-set of a graph can be decomposed into 1-factors is NP-complete, even given a decomposition of the graph's complement. The same is true of decomposition into triangles. Combinatorial applications of these results are also discussed.

1. PRELIMINARIES

A graph G is *decomposable* into a graph H, denoted H | G, if the edge set of G can be partitioned into sets, each inducing a graph isomorphic to H (for undefined terms, see [1]). Many researchers have searched for necessary and sufficient conditions on graphs G, H for H | G. Simply asking when $K_k \mid K_v$, over five hundred papers have been written since 1844; such decompositions are *Steiner systems* S(2,k,v) [6]. Many other restricted decomposition problems have also been studied [2].

Determining whether H | G in general appears to be computationally difficult. In fact, Holyer [7] shows that deciding whether $K_s \mid G$ is NP-complete for every $s \geq 3$. Holyer [8] also shows that deciding, for a 2n-vertex graph G, whether $nK_2 \mid G$ is NP-complete; moreover, it remains NP-complete even if G is required to be cubic. Another complexity result on graph decomposition shows that, for G an arbitrary n-vertex graph with n(n-1)/4 edges, deciding whether $G \mid K_n$ is polynomial time equivalent to the graph isomorphism problem [3,4].

In this paper, we are concerned with *completion problems*: deciding whether H | G given the additional information that $H \mid \overline{G}$ and $H \mid K_n$, for an n-vertex graph G. We consider the case when H is a 1-factor (a spanning subgraph which is regular of degree one), and the case when $H=K_3$. In both cases, we show that the completion problem is NP-complete. In the latter part of the paper, we describe some applications of these results in combinatorics.

2. COMPLETION BY 1-FACTORS

Let G be an n-vertex regular graph of degree r. A *Latin background* for G, denoted B(G;m,s), is an s by s symmetric array with elements chosen from {1,2,...,m}. Each diagonal entry contains the element m. In the first n rows, each position is either empty, or contains an element from the set

$\{r+1,r+2,\ldots,m\}$. In the latter $s-n$ rows, each position contains an element from $\{1,2,\ldots,m\}$. Every element appears at most once in each row (column). Finally, the pattern of empty positions is precisely an adjacency matrix for the graph G -- hence the term *background*.

The intuition here is that a Latin background $B(G;m,m)$ is "almost" a Latin square of order m. The only failing is that each of the first n rows has r empty positions, and each is missing the elements $\{1,\ldots,r\}$. Our first task is to construct Latin backgrounds.

Observation 2.1: For any n-vertex regular graph G, there is a $B(G;m,n)$ for any $m \geq 2n$.

Proof:

Let $A = (a_{ij})$ be an adjacency matrix for G. The entries b_{ij} of the background are as follows. On the diagonal, $b_{ii}=m$ for $1 \leq i \leq n$. Whenever $a_{ij} = 1$, the position b_{ij} is empty. In the remaining case, we set $b_{ij} = ((i+j) \bmod n) + n$. □

Lemma 2.2: Let L be a $B(G;m,s)$ with m even, for G an n-vertex regular graph of degree r. Denote by $P(i)$ the number of occurrences of element i in L. Let $N(i) = P(i)$ for $i > r$, and $N(i) = P(i) + n$ for $i \leq r$. Then there is a $B(G;m,s+1)$ if, for all $1 \leq i \leq m$, $N(i) \geq 2s-m$.

Proof:

Let T_i be the set of elements appearing in the i'th row of L. For $1 \leq i \leq n$, let $S_i = \{r+1,\ldots,m\} - T_i$; for $n < i \leq s$, let $S_i = \{1,\ldots,m\} - T_i$. Let $\{i_1,\ldots,i_k\}$ be the set of all elements for which $N(i_j) = 2s-m$. Using a proof of Cruse [5], we obtain that there is a system of distinct representatives $a_1, \ldots, a_s$ for $S_1, \ldots, S_s$ which contains all of the elements $i_1, \ldots, i_k$. We construct the required $B(G;m,s+1)$ by using the system of distinct representatives to "add a border" to the $B(G;m,s)$, and finally placing m in position $(s+1,s+1)$. □

Using observation 2.1 to construct an initial Latin background, and lemma 2.2 to repeatedly add a border to it, we obtain

Theorem 2.3: For any n-vertex regular graph G, there is a Latin background $B(G;m,m)$ for every even $m \geq 2n$. □

From a computational viewpoint, the proof of lemma 2.2 allows us to produce Latin backgrounds in polynomial time, by repeated use of the weighted bipartite matching algorithm [9] to find systems of the required type.

Consider a Latin background $B(G;m,m)$ for an r-regular n-vertex graph G.

The positions of the square occupied by the elements $\{1,\ldots,r\}$ comprise the adjacency matrix of an (m-n)-vertex r-regular graph H. The squares of the background occupied by the elements $\{r+1,\ldots,m-1\}$ comprise the adjacency matrix of $\overline{G \cup H}$. In fact, the actual positioning of these elements defines the decomposition into 1-factors. Specifically, the squares occupied by a fixed element form the adjacency matrix of one 1-factor. These observations about the meaning of Latin backgrounds enable us to prove that

Theorem 2.4: Deciding whether a graph has a decomposition into 1-factors is NP-complete, even if we are given such a decomposition of its complement.

Proof:

Membership in NP is immediate. To prove completeness, observe first that a cubic graph is 3-edge-colourable if and only if it can be decomposed into three 1-factors; by Holyer's theorem [8], this is NP-complete. The reduction proceeds by taking an arbitrary cubic graph G, and constructing B(G;2n,2n). This defines (as we have seen) a 3-edge-colourable cubic graph H such that $\overline{G \cup H}$ is decomposable into 1-factors; moreover, the computation is polynomial time. Now $G \cup H$ is decomposable if and only if G is 3-edge-colourable; this completes the proof. □

3. COMPLETION BY TRIANGLES

In combinatorial design theory, it is well-known that $K_3 \mid K_v$ if and only if $v \equiv 1$ or 3 (mod 6), and such decompositions are called *Steiner triple systems* [6]. In this section, we adapt a construction for Steiner triple systems from Latin squares to show that completion by triangles is also NP-complete.

Initially, we modify the construction of Latin backgrounds B(G;m,m) to produce a related structure called a *diagonal Latin background* DB(G;m-1,m-1). The definition is similar, with the one change that the element m appearing on the diagonal of a B(G;m,m) is replaced so that each element of $\{1,\ldots,m-1\}$ appears once on the diagonal. Constructing a DB(G;m-1,m-1) is easy, given a B(G;m,m).

Theorem 3.1: For an arbitrary graph G, deciding whether $K_3 \mid G$ is NP-complete even given a decomposition of $\overline{G}$ into triangles.

Proof:

Given an arbitrary n-vertex cubic graph G, construct both a DB(G;2n-1, 2n-1) and a DB(G;2n+1,2n+1). At least one of 2n-1 or 2n+1 is congruent to 1 or 3 (mod 6); let v be one of them which is. We construct a graph $\overline{H}$, and a decomposition of it into triangles. $\overline{H}$ has 2v+1 vertices $\{x_1, \ldots, x_v, y_1, \ldots, y_v, z\}$. Since $v \equiv 1$ or 3 (mod 6), there is a Steiner triple system of order v; we place a copy of it on the vertices $\{x_i\}$. Next we include the triangles $\{\{z,x_i,y_i\} \mid 1\le i\le v\}$. Lastly, if the square (i,j) $(i \ne j)$ of the DB(G;v,v) is not empty, but rather contains k, we include the triangle $\{x_k,y_i,y_j\}$. It is now easy to check that H, the complement of this graph, is decomposable into triangles if and only if G is 3-edge-colourable. This completes the proof. □

This provides a new proof of Holyer's NP-completeness result on decomposition into triangles. One interesting feature of both proofs is their use of the tools of combinatorial design theory -- notably, Latin squares and Steiner triple systems.

4. APPLICATIONS

In combinatorics and universal algebra, questions concerning the completion of partial algebras have been widely studied [10]; much of this activity has concentrated on quasigroups (= Latin squares). Many researchers have attempted to find necessary and sufficient conditions for a quasigroup to be completable. Theorem 2.4, restated in quasigroup vernacular, says that deciding whether a partial commutative quasigroup can be completed is NP-complete; this limits the potential of characterizing completable quasigroups.

Quasigroups are used in combinatorics to construct Steiner triple systems, which arise in the theory of error-correcting codes and experimental design theory, among others. The completion of partial Steiner triple systems has also been studied [10]. Theorem 3.1 shows that this problem is also NP-complete. Quasigroups, in the guise of "1-factorizations", also arise in combinatorial scheduling theory. Rosa and Wallis [11] introduce the notion of a *premature* set of 1-factors to describe partial round-robin tournaments which cannot be completed. In this context, theorem 2.4 shows that deciding prematurity is NP-complete.

We expect that other applications of these results exist, in the many areas where combinatorial designs arise.

ACKNOWLEDGEMENTS

Marlene Colbourn, Kevin Phelps, and Alex Rosa contributed much to the work reported here. Financial support was received from NSERC Canada Grant A5407.

REFERENCES

1. J.A. Bondy and U.S.R. Murty, *Graph Theory with Applications*, Macmillan, London, 1976.

2. F.R.K. Chung and R.L. Graham, "Recent results in graph decompositions", in *Combinatorics* (H.N.V. Temperley, editor) Cambridge University Press, London, 1981, pp. 103-121.

3. C.J. Colbourn and M.J. Colbourn, "Isomorphism problems involving self-complementary graphs and tournaments", *Proceedings of the Eighth Manitoba Conference on Numerical Mathematics and Computing*, Congressus Numerantium, Winnipeg, 1978, pp. 153-164.

4. M.J. Colbourn and C.J. Colbourn, "Graph isomorphism and self-complementary graphs", *SIGACT News 10* (1978) 25-29.

5. A.B. Cruse, "On embedding incomplete symmetric Latin squares", *Journal of Combinatorial Theory A16* (1974) 18-22.

6. J. Doyen and A. Rosa, "An updated bibliography and survey of Steiner systems", *Annals of Discrete Mathematics 7* (1980) 317-349.

7. I. Holyer, "The NP-completeness of some edge-partition problems", *SIAM Journal on Computing*, to appear.

8. I. Holyer, "The NP-completeness of edge-coloring", *SIAM Journal on Computing*, to appear.

9. J.E. Hopcroft and R.M. Karp, "An $n^{5/2}$ algorithm for maximum matching in bipartite graphs", *SIAM Journal on Computing 2* (1973) 225-231.

10. C.C. Lindner, "A survey of embedding theorems for Steiner systems", *Annals of Discrete Mathematics 7* (1980) 175-202.

11. A. Rosa and W.D. Wallis, "Premature sets of 1-factors", submitted for publication.

SHIFTING ALGORITHMS FOR TREE PARTITIONING
WITH GENERAL WEIGHTING FUNCTIONS

RONALD I. BECKER
(Department of Mathematics, University of Capetown, South Africa,
and University of Tennessee, Knoxville)

YEHOSHUA PERL
(Department of Mathematics and Computer Science, Bar-Ilan
University, Israel and Queens College, CUNY)

1. INTRODUCTION

In two recent papers [1], [3] we designed two shifting algorithms for optimum partitioning of a tree. Let $T = (V,E)$ be an (unrooted) tree with n edges. We associate a non-negative weight $w(v)$ with every vertex $v \in V$. A q-partition of T into q connected components $T_1, T_2, \ldots, T_q$ is obtained by deleting $k = q - 1$ edges of T, $1 < k < n$. We will also speak of a deleted edge as one having a "cut". The weight $W(T_i)$ of a component T_i is then the sum of the weights of its vertices.

The following two problems are considered:

Max-min q-partition: find a q-partition of T maximizing $\min_{1 \le i \le q} W(T_i)$.

Min-max q-partition: find a q-partition of T minimizing $\max_{1 \le i \le q} W(T_1)$.

In [3] we presented a shifting algorithm for finding a max-min q-partion of T. A more complicated shifting algorithm for finding a min-max q-partition of T is given in [1].

In both algorithms an arbitrary terminal vertex is chosen as the root of T. All k cuts are initially assigned to the edge incident with the root. Then shifts of cuts from an edge to an adjacent edge are applied to improve the current partition. The first algorithm [3] requires only down shifts from an edge to a son edge. The second algorithm [1], [2] requires down shifts (according to another criterion) and sideshifts from an edge to a brother edge having a correction effect. The complexities of the two algorithms are $O(k^2 \cdot \text{radius}(T) + kn)$ and $O(k^3 \cdot \text{radius}(T) + kn)$, respectively.

In the present paper we generalize the definition of the weight $W(T_i)$ of a component. We seek conditions on W which ensure that the max-min or min-max problem remains solvable by the algorithms in [3] and [1] when the weighting function given at the start of this introduction is replaced by W. We will describe two classes of function W. The first, the basic weighting functions are such that the algorithm of [3] solves the max-min problem for this class. The second, the invariant weighting functions, are the basic weighting functions with a further restriction. The algorithm of [1] solves the min-max problem for these weighting functions. An example of a weighting function which is basic but not invariant is the diameter. We show, in addition, that a nontrivial modification of the algorithm of [1] solves the min-max diameter problem.

2. BASIC WEIGHTING FUNCTIONS

We define a weighting function on a tree T to be an assignment of a non-negative real number to each subtree of T. A basic weighting function is a weighting function satisfying

(B_1) If T_1 and T_2 are subtrees of T and if T_1 is a subtree of T_2 then $W(T_1) \leq W(T_2)$.

Examples of basic weighting functions are:

1) $W(T') = \sum_{\substack{v \in T' \\ v \text{ a vertex}}} w(v)$. Max-min problems for this weighting function are considered in Perl and Schach [3].
2) $W(T')$ = diameter of T'.
3) Choose a fixed root r of T. Define $W(T')$ = height of T'.
4) $W(T')$ = sum of the lengths of the edges of T'.
5) $W(T')$ = sum of the lengths of the edges and weights of the vertices of T'.
6) $W(T')$ = sum of the lengths of all paths between the vertices of T'.
7) $W(T')$ = maximum vertex weight over vertices of T', OR $W(T')$ = minimum edge length over edges of T'. (The min-max and max-min problems associated with these are trivial).

A component is a subtree and so any weighting function assigns weights to components. If edges have length and so also contribute to the weight, it must be decided what to do with the deleted edges (see definition of q-partition). We choose to include such an edge in the component immediately below it with respect to the root (but other choices can also be made).

Algorithm 1 (Max-min q-partition of a Tree:Basic weighting functions)

1) Initially place all k cuts on the edge incident with the root.

2) At every step, consider each of the possible shifts of every cut, and compute the weight of the down-component that would result from each new shift. Perform the shift of a cut that maximizes the weight of the resulting down-component, unless the latter is lighter than the current lightest component, in which case terminate. In case of ties, an arbitrary choice may be made.

THEOREM 1

Algorithm 1 finds a max-min q-partition of a tree for all basic weighting functions.

NOTE: The proofs of this theorem and of theorems 2 and 3 are quite complicated, and are omitted here.

3. INVARIANT WEIGHTING FUNCTIONS

A basic weighting function is invariant if it satisfies the following condition:

(B_2) Let e and e' be two edges emanating from a vertex v of the rooted tree T, and let T_1 be the subtree at v consisting of the son of v at the end of e and all its descendents; T_2 the subtree rooted at v consisting of the son of v at the end of e' and all its descendents. Let T' be a subtree containing v as terminal vertex. Then $W(T_1) \leq W(T_2)$ implies

$$W(T' \cup T_1) \leq W(T' \cup T_2).$$

Examples of invariant weighting functions are examples 1, 3, 5, 7 and 8 of section 2 (on basic weighting functions). The diameter function (example 2) and also the weighting function of example 6 are however not invariant. This is illustrated in Fig. 1 ((a) for diameter and (b) for example 6).

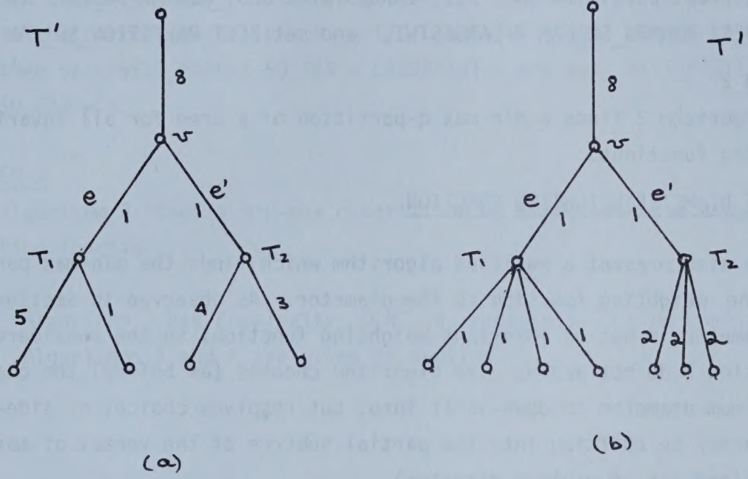


Figure (a) $14 = D(T' \cup T_1) > D(T' \cup T_2) = 13$

$6 = D(T_1) < D(T_2) = 7$

(b) $70 = W(T' \cup T_1) > W(T' \cup T_2) = 66$

$13 = W(T_1) < W(T_2) = 16$

Algorithm 2 (Min-max q-partition of a Tree: Invariant Weighting Functions)

1. Place all k cuts on the edge incident with the root. Set BEST_MINMAX_SO_FAR ← ∞ , and set BEST_PARTITION_SO_FAR equal to the starting configuration.
2. While the root component is not a heaviest component, perform steps 3, 4 and 5 .
3. Find a cut with a heaviest down-component, and down-shift it from its current edge e to a vacant son-edge having heaviest down-component. If no such vacant edge exists then halt.
4. Traverse the path from tail(e) to the root in the bottom-up direction until a vertex v which is the head of a cut is encountered. For each vertex w on that path having a cut incident from w , perform the following:

    If the down-component of a cut incident from w is lighter than the down-component of the vacant son-edge e_s of w on the path, then side-shift that cut to edge e_s. If more than one cut incident from w can be side-shifted, choose a cut with a lightest down-component.

5. Set LARGESTWT equal to the weight of the largest component in the current partition A . If LARGESTWT < BEST_MINMAX_SO_FAR then set BEST_MINMAX_SO_FAR ← LARGESTWT, and set BEST_PARTITION_SO_FAR ← A .

THEOREM 2

Algorithm 2 finds a min-max q-partition of a tree for all invariant weighting functions.

4. THE DIAMETER WEIGHTING FUNCTION

We also present a modified algorithm which finds the min-max partition when the weighting function is the diameter. As observed in Section 3, the diameter is not an invariant weighting function, so the considerations of Section 3 do not apply. The algorithm chooses (as before) the component of maximum diameter to down-shift into, but resolves choices or side-shifts at a vertex by shifting into the partial subtree of the vertex of maximum height (and not of maximum diameter).

Algorithm 3 (Min-max Diameter q-partition of a Tree)

1. Place all k cuts on the edge incident with the root. Set BEST_MINMAX_SO_FAR ← ∞, and set BEST_PARTITION_SO_FAR equal to the starting configuration.

2. While the root component is not a component of largest diameter, perform steps 3,4 and 5.

3. Find a cut with largest diameter down-domponent, and down-shift it from its current edge e to a vacant son-edge. If there is a choice, shift to a down-component of largest height. If no vacant son-edge exists, then halt.

4. Traverse the path from tail(e) to the root in the bottom-up direction until a vertex v which is the head of a cut is encountered. For each vertex w on the path having a cut incident from w perform the following:

If the down-component of a cut incident from w is of smaller height than the down-component of the vacant son-edge e_s on the path then side-shift that cut to e_s . If more than one cut incident from w can be side-shifted, choose a cut with down-component of smallest height.

5. Set LARGESTWT equal to the weight of the component of largest diameter in the current partition A . If LARGESTWT < BEST_MINMAX_SO_FAR then set BEST_MINMAX_SO_FAR ← LARGESTWT , and set BEST_PARTITION_SO_FAR ← A.

THEOREM 3

Algorithm 3 finds a min-max q-partition of a tree for the diameter weighting function.

NOTE: Algorithm 3 has complexity $O(k^2. n. \text{radius}(T))$. (The complexity of Algorithms 1 and 2 are given in section 1).

REFERENCES

[1] Becker, R.I., PERL, Y. and Schach, S. A shifting algorithm for min-max tree-partitioning. To appear, J.A.C.H.

[2] Becker, R.I., Perl, Y. and Schach, S. A shifting algorithm for min-max tree partitioning. Proc. 7th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, Springer-Verlag, Berlin 1980, 64-75.

[3] Perl, Y. and Schach, S. Min-max tree partitioning. J.A.C.M. 28(1981), 5-15.

CUTWIDTH PROBLEMS IN GRAPHS

E. M. GURARI
Department of Computer Science
SUNY at Buffalo
Amherst, N.Y. 14226

and

I. H. SUDBOROUGH
Electrical Engineering and Computer Science Dept.
Northwestern University
Evanston, Illinois 60201

ABSTRACT

It is shown that the problem of deciding if a layout exists for a finite graph G=(V,E) which makes it have "cutwidth" k, for each fixed value of k, is solvable deterministically in $O(n^k)$ steps and is solvable nondeterministically in log n space. The complexity of several problems under cutwidth constraints is also discussed. For example, the GRAPH ACCESSIBILITY problem, the AND/OR GRAPH ACCESSIBILITY problem, and the 3COLOR problem for graphs which are restricted to have cutwidth bounded by some function f are considered.

INTRODUCTION

Let G=(V,E) be a finite undirected graph. A one-to-one function $\ell : V \longrightarrow \{1,\ldots,|V|\}$ is a *layout* of G. Let k be a positive integer. G has *cutwidth* k under the layout ℓ, denoted by $cutwidth_{\ell}(G)=k$, if, for all i ($1 \le i < |V|$), $|\, \{\{u,v\} \mid \ell(u) \le i < \ell(v)\} \,| \le k$. For example, the graph shown in Figure 1 has cutwidth 3 under the layout shown. The *cutwidth of G*, denoted by *cutwidth*(G), is $\min\{\, cutwidth_{\ell}(G) \mid \ell \text{ is a layout of } G\}$. The problem of deciding whether a layout ℓ exists such that $cutwidth_{\ell}(G)=k$, for an arbitrary given positive integer k, which is called the MINIMUM CUT LINEAR ARRANGEMENT problem [1], is

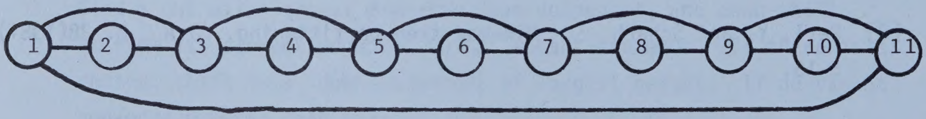


Figure 1. This graph has cutwidth 3 under the given layout.

known to be $\mathbb{NP}$-complete [2]. On the other hand, a linear time algorithm for determining an optimal cutwidth layout for a complete m-ary tree and a polynomial time algorithm for finding a layout for an arbitrary tree that makes the cutwidth no more than twice the optimal are also known [3].

A graph $G=(V,E)$ has *bandwidth* k under a layout $\ell : V \xrightarrow{1-1} \{1,\ldots,|V|\}$, denoted by *bandwidth* $_{\ell}(G)=k$, if, for all $\{x,y\} \in E$, $|\ell(x)-\ell(y)| \leq k$. The *bandwidth of G*, denoted by *bandwidth*(G), is min $\{$*bandwidth* $_{\ell}(G) \mid \ell$ is a layout of $G\}$. The problem of deciding whether there exists a layout that makes a given graph G have bandwidth k, when k is an input quantity, which is called the BANDWIDTH MINIMIZATION PROBLEM, is known to be $\mathbb{NP}$-complete [4]. Furthermore, it is known that the problem of deciding if there exists a layout ℓ for a given graph G such that *bandwidth* $_{\ell}(G)=k$, for each *fixed* value of k, can be solved by a deterministic algorithm in $O(n^{k})$ steps, where n is the number of vertices in the graph G [5]. (This is based on a modification of a dynamic programming algorithm of Saxe [6].) In the case $k=2$, it is known that deciding if there is a layout ℓ such that *bandwidth* $_{\ell}(G)=2$ is solvable in linear time [7]. It should, perhaps, be noted that the notions of bandwidth and cutwidth defined above can be easily extended to directed graphs.

We indicate briefly some of the known relationships between bandwidth and cutwidth of graphs. First, we note that a bandwidth restriction on a graph implies a cutwidth restriction. Let $\ell : V \xrightarrow{1-1} \{1,\ldots,|V|\}$ be a layout of a graph such that *bandwidth* $_{\ell}(G)=k$, then *cutwidth* $_{\ell}(G) \leq k(k+1)/2$. Furthermore, if G has degree d, for some $d>0$, then *cutwidth* $_{\ell}(G) \leq d \cdot$ *bandwidth* $_{\ell}(G)$. Secondly, we show that there are graphs with cutwidth 2 which have arbitrarily large bandwidth. Therefore, a cutwidth restriction does not imply a bandwidth restriction. We describe now a collection of graphs $\{H_n\}_{n\geq 1}$ such that, for all n, H_n has cutwidth 2 and bandwidth at least $(n+1)/4$. Define H_n as follows: H_n contains n rows of vertices. The i-th row consists of i vertices and there is an edge connecting the j-th vertex in row i with the (j+1)-st vertex in row i, for all j ($1 \leq j < i$). There is also an edge connecting the i-th vertex in row i with the (i+1)-st vertex in row i+1, for all i ($1 \leq i < n$). The graph H_4 is shown in Figure 2. It is straightforward to verify that

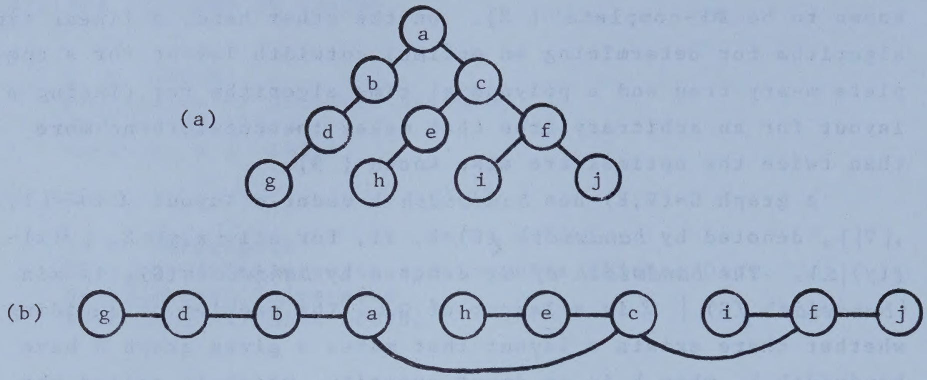


Figure 2 (a) The graph H_4 (b) a layout of H_4 with cutwidth 2.

for all $n \geq 1$, there is a layout ℓ such that *cutwidth*$_\ell(H_n)=2$. Such a layout ℓ just assigns the vertices of each row to consecutive integers and assigns consecutive rows to consecutive sets of integers. For example, the graph H_4 is layed out with cutwidth 2 in Figure 2(b) above. On the other hand, the graph H_n has bandwidth greater than $(n+1)/4$. This follows from the well-known fact that the bandwidth of a graph must be at least as large as the size of the graph divided by its diameter. H_n has $n(n+1)/2$ vertices and diameter $2(n-1)$. Thus, *bandwidth*$(H_n) \geq n(n+1)/4(n-1) > (n+1)/4$.

ALGORITHMS FOR TESTING CUTWIDTH(G)=k

A *partial layout* of a graph $G=(V,E)$ is a one-to-one function ℓ from some subset V' of V to the set of integers $\{1,\ldots,|V'|\}$. An edge $\{u,v\}$ is *dangling* from ℓ, if $u \in$ *domain*(ℓ) and $v \notin$ *domain*(ℓ). The set of all edges dangling from ℓ is denoted by *dangling*(ℓ). The partial layout ℓ is *k-plausible* (for the *cutwidth*$(G) \leq k$ problem) if (1) *cutwidth* $(G) \leq k$ and (2) $|$*dangling* $(\ell)| \leq k$. (The reader should note that *cutwidth* $(G) \leq k$ means, when ℓ is a partial layout, that $|\{\{u,v\} \mid \ell(u) \leq i < \ell(v)$ and u,v are both in *domain*$(\ell)\}| \leq k$, for all i ($1 \leq i < |$*domain*$(\ell)|$).) The set of *active nodes* of the partial layout ℓ, denoted by *active*(ℓ), is the set $\{u \in$ *domain*$(\ell) \mid \{u,v\} \in$ *dangling*$(\ell)\}$. Two k-plausible partial layouts ℓ_1 and ℓ_2 are *k-equivalent* if (1) *active*$(\ell_1)=$*active*(ℓ_2) and (2) *dangling*$(\ell_1)=$ *dangling*(ℓ_2).

Remark: In what follows only *connected* graphs of degree>1 are considered. The results can easily be generalized to the unrestricted case.

The following lemma is proven in the same way that was used for the corresponding result about bandwidth in [6]:

Lemma 1. Let ℓ_1 and ℓ_2 be partial layouts of a graph G. ℓ_1 and ℓ_2 are k-equivalent if and only if *domain*(ℓ_1)=*domain*(ℓ_2).

Lemma 1 indicates that the pair (*active*(ℓ),*dangling*(ℓ)) is sufficient to deduce *domain*(ℓ). Let us say that a partial layout ℓ_2 is a *successor* of a partial layout ℓ_1 if ℓ_2 is k-plausible and extends ℓ_1 by exactly one element. We shall also say that the pair (*active*(ℓ_2),*dangling*(ℓ_2)) is a successor of the pair (*active*(ℓ_1),*dangling*(ℓ_1)).

Theorem 1. The cutwidth(G)$\leq$k problem is solvable (1) nondeterministically in logarithm space and (2) deterministically in O(n^k) steps, where n is the number of vertices in the graph G.

Proof: Let G=(V,E) be any given graph. The following nondeterministic algorithm checks whether cutwidth(G)$\leq$k:

Step 1. Choose nondeterministically a layout ℓ_0 consisting of one node, say v. Then set the program variable p to the pair (active(ℓ_0),dangling(ℓ_0)).

Step 2. Assume that p contains the pair (active(f), dangling(f)), for some k-plausible partial layout f of G. Then, nondeterministically choose a vertex x and verify that it is not in the domain of f. This can be done by checking that there is a path from one of the active nodes to this node x that passes through one dangling edge. That such a path question can be done in logarithm space follows from the observations by Savitch on the graph accessibility problem [8]. Let f' be the extension of the layout f by adding the node x and assigning it the next available integer.

Step 3. If dangling(f') is the empty set, then stop and assert that G has cutwidth k. Otherwise, set p to the pair (active(f'),dangling(f')) and return to Step 2. (It should, perhaps, be mentioned that if at any time there are more than k dangling edges, then the algorithm terminates unsuccessfully.)

This nondeterministic algorithm needs at most log n worktape space to represent the program variable p, since p always contains a pair (active(f),dangling(f)), for some k-plausible partial layout f, and therefore can be represented by writing the index of each of the edges and vertices involved (in binary notation).

The deterministic algorithm for the cutwidth(G)$\le$k problem uses two data structures:

(1) a queue Q whose elements are pairs (a,d), where a= active(f) and d=dangling(f), for some k-plausible partial layout f, and

(2) an array A which contains one element for each possible pair p=(a,d), where a=active(f) and d=dangling(f) for some k-plausible partial layout f. Each element A[p] of A consists of a Boolean value telling whether the pair p=(a,d) has already been considered or not.

At the beginning Q contains the pair $\Phi=(\phi,\phi)$ corresponding to the partial layout f whose domain is empty. The A[p]'s are initially false. The algorithm is the following:

Step 1. Extract a pair p=(a,d) from Q.

Step 2. Determine the successors of p=(a,d).

Step 3. For each successor s of p=(a,d) such that A[s] is false perform the following steps:

(a) Set A[s] to true.

(b) If there are no dangling edges in s, then stop asserting that cutwidth(G)$\le$k.

(c) Insert s at the end of Q.

Step 4. If Q is empty, then halt asserting that cutwidth (G)>k. Otherwise, proceed to Step 1.

To determine the running time of this algorithm we note that: (1) for each pair (a,d), where d is a set of k edges, there are at most k successors, since the next node must be one of the nodes at the end of a dangling edge. (Otherwise, since the next node is connected, there would be a cut of size at least k+1 after this node.) Moreover, there are O(n^k) distinct pairs (a,d), where d is a set of k edges, since there are $\binom{|E|}{k}$ such sets d and $|E|=\#$ of edges$\le 2k|V|=2kn$. Therefore, there are $\binom{2kn}{k}$ or O(n^k) possible sets d, and (2) for each pair (a,d), where d is a set of k-1 or fewer edges, there can be at most n successors. These successors can be obtained in O(n) steps by traversing edges starting with a dangling edge and not passing through another dangling edge. There are $O(n^{k-1})$ distinct pairs (a,d), where d is a set of at most k-1 edges.

Thus, the total amount of time needed by the algorithm is $O(n^{k})\cdot k + O(n^{k-1})\cdot n = O(n^{k})$. ☐

We observe without proof that the problem of testing isomorphism of two graphs which have cutwidth k can be done in $O(n^{2k})$ steps by an algorithm similar to the one presented above. The details of this algorithm are contained in [11].

THE GRAPH ACCESSIBILITY PROBLEM

The graph accessibility problem (GAP) is the problem of deciding if a path exists in a given directed graph from a specified start node to a specified goal node. GAP is log space complete for $\mathbb{N}SPACE(\log n)$ and is in $\mathbb{D}SPACE(\log^2 n)$ [8]. These facts form the foundation of much recent work on the relationship between nondeterministic and deterministic space bounded complexity classes and, in particular, Savitch's result that $\mathbb{N}SPACE(S(n)) \subseteq \mathbb{D}SPACE(S(n)^2)$, for all $S(n) \geq \log n$ [8]. In [12] an extension of Savitch's theorem was reported in which GAP was restricted to graphs of bandwidth bounded by some function f(n). Let $GAP_{bandwidth}(f(n))$ denote GAP restricted to graphs G=(V,E) with bandwidth $f(|V|)$. That is,

$GAP_{bandwidth}(f(n))$:

Instance: A graph G=(V,E) with a layout $\ell: V \xrightarrow{1-1} \{1,\dots,|V|\}$

Question: (1) Is $bandwidth_{\ell}(G) \leq f(|V|)$?

(2) Does G have a path from its designated start node to its designated goal node?

It was shown in [12] that $\{GAP_{bandwidth}(f^{k}(n))\}_{k\geq 1}$ is a family that is log space complete for $\mathbb{N}SPACE(\log f(n))$ and that $GAP_{bandwidth}(f(n))$ is in $\mathbb{D}SPACE(\log f(n) \times \log n)$, for $f(n) \leq n$. As noted in [12], it follows that $\mathbb{N}SPACE(S(n)) \subseteq \mathbb{D}SPACE(S(n) \times \log n)$, for $S(n) \leq \log n$ [12]. We show that $GAP_{cutwidth}(f(n))$ is log space reducible to $GAP_{bandwidth}(f(n))$ and, therefore, we derive the result that $GAP_{cutwidth}(f(n)) \in \mathbb{D}SPACE(\log f(n) \times \log n)$. The problem $GAP_{cutwidth}(f(n))$ is defined by replacing each occurrence of the word bandwidth in the above by cutwidth.

Lemma 2. $GAP_{cutwidth}(f(n)) \le_{log} GAP_{bandwidth}(2 \cdot f(n))$, for all functions f such that $f(n) \le n$.

Proof: Let $G=(V,E)$ be a finite directed graph with a designated start node s and a designated goal node g. Let ℓ be a layout of G such that $cutwidth_{\ell}(G)=k$. This means that at most k edges connect vertices x and y such that $\ell(x) \le i < \ell(y)$, for each i ($1 \le i \le |V|$). For each i ($1 \le i \le |V|$), let E_i be the set $\{e^{(i)} \mid e=(x,y) \varepsilon E$ for some $x,y \varepsilon V$ such that $\ell(x) \le i < \ell(y)\} \cup \{ e^{(i)} \mid e=(x,y) \varepsilon E$ for some $x,y \varepsilon V$ such that $\ell(x) > i \ge \ell(y) \}$. Since G has cutwidth k under the layout ℓ each set E_i has cardinality at most k. Construct the graph $G'=(V',E')$, where:

$$V' = V \cup (\bigcup_{1 \le i \le |V|} E_i)$$

$$\begin{aligned} E' = & \{(e^{(i)}, e^{(i+1)}) \mid e=(x,y) \text{ and } \ell(x) \le i < \ell(y) \} \\ & \cup \{(e^{(i+1)}, e^{(i)}) \mid e=(x,y) \text{ and } \ell(x) > i \ge \ell(y) \} \\ & \cup \{(e^{(\ell(y))}, y) \mid e=(x,y) \text{ and } \ell(y) < \ell(x) \} \\ & \cup \{(x, e^{(\ell(x))}) \mid e=(x,y) \text{ and } \ell(x) < \ell(y) \} \\ & \cup \{(e^{(\ell(y)-1)}, y) \mid e=(x,y) \text{ and } \ell(x) < \ell(y) \} \\ & \cup \{(x, e^{(\ell(x)-1)}) \mid e=(x,y) \text{ and } \ell(y) < \ell(x) \} \end{aligned}$$

It is straightforward to verify that G has a path from s to g if and only if G' has a path from s to g. Therefore, G is in GAP if and only if G' is in GAP. Furthermore, G' can be layed out to have bandwidth at most 2k. That is, one defines a layout ℓ' such that the vertex $\ell^{-1}(i)$ and all of the vertices in E_i are assigned integers smaller than those assigned to $\ell^{-1}(i+1)$ and the vertices in E_{i+1}, for all i ($1 \le i < |V|$). This transformation and layout is shown in Figure 3. It follows, in any case, that under this layout ℓ', for all $(u,v) \varepsilon E$, $|\ell'(u) - \ell'(v)| \le 2k$. Thus, G' has bandwidth 2k. Since the transformation from G to G' is easily computed within log n space (as well as the layout of G'), it follows that $GAP_{cutwidth}(f(n)) \le_{log} GAP_{bandwidth}(f(n))$. □

Lemma 2 extends the applicability of the divide-and-conquer algorithm given in [12] to a wider class of graphs, since we have seen that there are graphs that have arbitrarily large bandwidth but cutwidth 2.

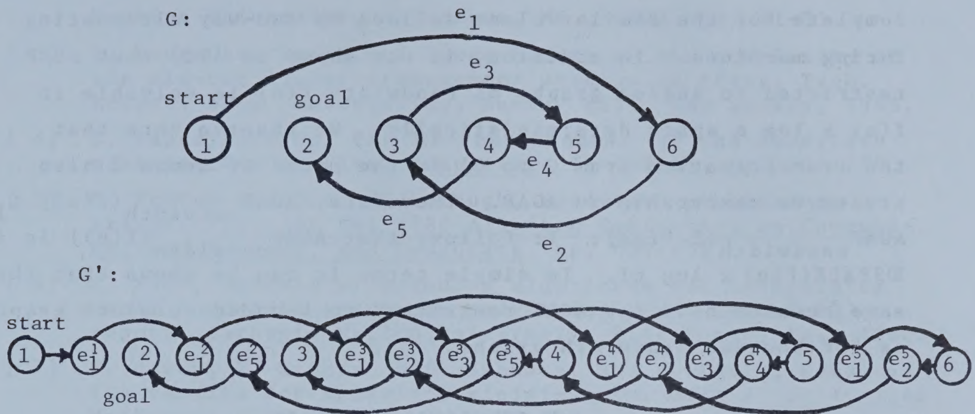


Figure 3. The transformation described in the proof of Lemma 2 and the layout described for the graph G'.

THE AND/OR GRAPH ACCESSIBILITY PROBLEM

An and/or graph is a directed graph $G=(V,E)$ together with a labeling function $label: V \rightarrow \{\text{and}, \text{or}\}$. One can play the following pebble game on an and/or graph $G=(V,E,label)$: (a) one can always place a pebble on a vertex with no predecessors, (b) if x is a vertex and $label(x)=\text{and}$, then a pebble can be placed on x only if every predecessor of x contains a pebble, and (c) if x is a vertex and $label(x)=\text{or}$, then a pebble can be placed on x only if at least one predecessor of x contains a pebble. The AND/OR GRAPH ACCESSIBILITY problem (AGAP) is to decide whether a pebble can be placed on a designated goal node by playing this pebble game on a given and/or graph. It is known that the problem AGAP is log space complete for the complexity class $\mathbb{P}$ [9]. In [10] it was shown that the problem AGAP restricted to monotone and/or graphs of bandwidth f(n) is log space complete for the complexity class TIME-SPACE(poly,f(n)). (The class TIME-SPACE(poly,f(n)) represents those problems that can be solved by deterministic Turing machines in polynomial time and simultaneous f(n) space. An and/or graph is monotone under a layout ℓ if, for every $(x,y)\varepsilon E$, $\ell(x) \leq \ell(y)$.) It was also shown in [10] that the problem GAP restricted to monotone and/or graphs of bandwidth f(n) is log space complete for the class of problems accepted by one-way alternating Turing machines in log f(n) space and without the monotone condition it was complete for the similar class defined by two-way alternating Turing machines. In addition, it was shown in [10] that AGAP restricted to and/or graphs of bandwidth f(n) is solvable in f(n) x log n space deterministically. We observe here that the transformation from G to G' in the proof of Lemma 2 also preserves membership in AGAP. Therefore, $\mathrm{AGAP}_{\mathrm{cutwidth}}(f(n)) \leq_{\log} \mathrm{AGAP}_{\mathrm{bandwidth}}(2 \cdot f(n))$. It follows that $\mathrm{AGAP}_{\mathrm{cutwidth}}(f(n))$ is in DSPACE(f(n) x log n). In simple terms it can be shown that the same results hold for AGAP restricted to bounded cutwidth graphs as for bounded bandwidth graphs.

3-COLORATION PROBLEM

In [13] it was shown that several NP-complete problems when restricted to bandwidth f(n) were log space complete for the complexity class NONDETERMINISTIC-TIME-SPACE(poly,f(n)). (This class is often denoted by NTISP(poly,f(n)).) In particular, the 3-coloration problem for graphs restricted to bandwidth f(n) and node degree 5 is log space complete for NTISP (poly,f(n)). Since such graphs have cutwidth at most $5 \cdot f(n)$, it can be seen that the 3-coloration problem for graphs of cutwidth f(n) is also log space complete for NTISP(poly,f(n)). In particular, this means that the 3-color problem can be solved deterministically in polynomial time when the graph is restricted to one with cutwidth log n, since then the problem is in NTISP(poly,log n)=NSPACE(log n) and NSPACE(log n) is a subset of P. In fact, work has been conducted in [14] to show that several other NP-complete problems, when restricted to cutwidth f(n), are log space complete for NTISP(poly,f(n)). These problems include: 3SATISFIABILITY, VERTEX COVER, INDEPENDENT SET, DOMINATING SET, DOMATIC NEMBER, MONOCHROMATIC TRIANGLE, 3-DIMENSIONAL MATCHING, PARTITION INTO TRIANGLES, and EXACT COVER BY 3SETS. We shall not give further details here on account of space restrictions.

REFERENCES

[1] M.R. Garey and D.S. Johnson, Computers and Intractability: A guide to the theory of NP-completeness, W. Freeman and Co., San Francisco, 1979.

[2] F. Gavril, Some NP-complete problems on graphs, Proc. of the 11th Conf. on Info. Sciences and Systems, The Johns

Hopkins University, Baltimore, Md., 1977, pp. 91-95.

[3] T. Lengauer, Upper and lower bounds on the complexity of the min-cut linear arrangement problem on trees, Tech. Report, Bell Laboratories, Murray Hill, New Jersey, 1980.

[4] C. Papadimitriou, The ℕP-completeness of the bandwidth mimimization problem, Computing 16 (1976), pp. 263-270.

[5] B. Monien and I.H. Sudborough, Bandwidth problems in graphs, Proc. of the 1980 Allerton Conference on Communication, Control, and Computing, pp. 650-659.

[6] J. Saxe, Dynamic-programming algorithms for recognizing small bandwidth graphs in polynomial time, Technical report, Carnegie-Mellon University, Pittsburgh, Pa., 1980.

[7] M. Garey, R. Graham, D.S. Johnson, and D. Knuth, Complexity results for bandwidth minimization, SIAM J. on Applied Mathematics 34 (1978), pp. 477-495.

[8] W.J. Savitch, Relationships between nondeterministic and deterministic tape complexities, J. Computer and System Sciences 4,2 (1970), pp. 177-192.

[9] A.K. Chandra, D.C. Kozen, and L.J. Stockmeyer, Alternation, J. Assoc. Computing Mach. 28,1 (1981), pp. 114-133.

[10] I.H. Sudborough, Efficient algorithms for path system problems and applications to alternating and time-space complexity classes, Proc. 21st Annual IEEE Symp. on Foundations of Computing, (1980), pp. 62-73.

[11] E.M. Gurari and I.H. Sudborough, Bounded cutwidth graphs, Technical Report, Northwestern University, Evanston, Illinois, 60202.

[12] B. Monien and I.H. Sudborough, On eliminating nondeterminism from Turing machines that use less than logarithm worktape space, in Vol 72 Lecture Notes in Computer Science, Springer Verlag (1979), pp. 431-445.

[13] B. Monien and I.H. Sudborough, Bandwidth constrained ℕP-complete problems, Proc. 13th Annual ACM Theory of Computing Symp. (1981).

[14] M-J. Chung and I.H. Sudborough, work in progress.

POLYNOMIAL CONSTRUCTION OF OPTIMAL PREFIX TABLES FOR TEXT COMPRESSION

Y. Choueka(1), A. S. Fraenkel(2), Y. Perl(1)*

Abstract

Frequently occurring prefixes of the words in a text may be replaced by short numerical codes. Given a table of fixed size m for storing the set of such codes, heuristic methods for selecting an optimum set have been given in the literature previously. We present a dynamic programming algorithm with a look-ahead property of polynomial complexity for constructing an optimal set of prefixes of size m.

1. INTRODUCTION

The compression method studied here is based on the common observation that for most English texts, a small set of suffixes accounts for a large part of all the suffixes of words. Thus storage can be saved by replacing common suffixes of words by a pointer to a compression table in which these suffixes will be stored (see e.g., Gotlieb et al. [3] and Radhakrishnan [5]). If the code length is d bits, then the compression table can accomodate $m = 2^d$ suffixes, and the problem then is that of selecting a set of m suffixes which will maximize the compression.

The same ideas can also be applied for prefixes. This is especially efficient for languages such as Hebrew and Arabic, where prefixes abound. Since the problems for suffixes and prefixes can be reduced to one another by writing a word in reverse form, we shall henceforth restrict attention to the latter.

Obviously we assume in the sequel that if a word x has more than one prefix in the compression table S, then the longest prefix of x will be used for compressing x. This defines the encoding procedure uniquely for every word (which would not be the case if we were encoding general substrings rather than just prefixes, see e.g., Doucette et al. [1] and Rubin [6]).

In the next section a tree representation of the problem is presented. In Section 3 we show that a straightforward application of the dynamic programming technique does not give an optimal solution. In Section 4 we present first a valid dynamic programming procedure with a look-ahead property, and later on a polynomial step-by-step refinement of this procedure.

2. TREE REPRESENTATION OF THE PROBLEM

An ordered tree is a rooted tree in which, for every vertex v, the sons of v are ordered lexicographically from left to right.

A dictionary over an alphabet Σ of $|\Sigma| = t$ letters can be described by means of a t-ary ordered rooted tree T in the following way. Each edge emanating from a vertex is associated with one letter of Σ and each vertex v of T represents the string of letters obtained by concatenating the letters associated with the edges along the path from the root r to v. The given set of words is represented by a subset of the vertices of T.

---

(1) Department of Mathematics and Computer Science, Bar-Ilan University, Ramat-Gan, Israel.

(2) Department of Applied Mathematics, The Weizmann Institute of Science, Rehovot, Israel.

* On Sabbatical Leave at the Department of Computer Science, Queens College of the City University of New York.

Suppose for example that each letter is stored using one byte of 8 bits. If a compression table of size $m = 2^d$ is used for storing m prefixes, then for a prefix of ℓ letters we save in this compression $8\ell - d$ bits or $\ell - c$ bytes where $c = [d/8]$, since the prefix is replaced by a pointer of d bits (c bytes).

Let $\ell(v)$ and $f(v)$ denote the number of letters and the frequency of the string represented by vertex v of T. Thus $\ell(v)$ is the distance of v from the root r of T, also called the level of v. The weight $w(v)$ of a vertex v is defined as the sum of the frequencies of the vertices in the subtree rooted at v. The weights of all the vertices of T are computed by a linear endorder scanning of T, since the weight of a vertex is the sum of its frequency and the weights of its sons.

The compression--in bytes--obtained by inserting one prefix represented by a vertex v of T into the compression table is $(\ell(v) - c)w(v)$. However, the compression obtained for a set of prefixes is not necessarily the sum of the compressions obtained by each one of them separately. Let S be the subset of vertices of T representing the prefixes in the compression table. A vertex $v_1 \in S$ is called a predecessor of a vertex $v_2 \in S$ (and v_2 is called a successor of v_1), if there is a directed path in T from v_1 to v_2 containing no other vertices of S. Denote the set of successors of v by $SC(v)$ and the unique predecessor of v by $PR(v)$.

If $v_1 = PR(v_2)$, then by the basic encoding rule detailed above, none of the words represented by vertices in the subtree rooted at v_2 will be compressed using the prefix represented by v_1, since a better compression is obtained for them using the longer prefix represented by v_2 (or successors of v_2 or their successors, etc.). The prefix represented by v_1 compresses precisely all words represented by vertices in the subtree rooted at v_1, excluding those represented by vertices in the subtrees rooted at the successors of v_1. Thus the compression obtained by using the prefix represented by a vertex v is $(\ell(v) - c)(w(v) - \sum_{x \in SC(v)} w(x))$. Hence the compression $C(S)$ obtained by a subset S of the vertices of T is

$$(1) \qquad C(S) = \sum_{v \in S} (\ell(v) - c)(w(v) - \sum_{x \in SC(v)} w(x)).$$

A subset maximizing the compression over all subsets of m vertices is called an optimal subset. Our purpose is to compute efficiently an optimal subset S for a given tree T with given frequencies associated with the vertices.

3. HOW TO APPLY DYNAMIC PROGRAMMING TO OBTAIN OPTIMUM COMPRESSION

Using the tree representation of the problem, it is natural to apply a dynamic programming approach to find an optimal subset. In order to obtain an efficient dynamic programming procedure, it is customary to process the tree in bottom-up order. For each vertex v and each integer k, $k \leq m$, an optimal subset of k vertices of the subtree of T rooted at v is computed, using the optimal subsets of i vertices, $i \leq k$, for the subtrees rooted at the sons of v.

However, the following example demonstrates that this straightforward approach does not yield an optimal subset for T. Consider the tree T drawn in Figure 1, where the frequency f and the weight w of each vertex v appear to its left and right, respectively.

Assume $c = 1$. Consider the subtree rooted at vertex 4. It is easy to see that the optimal subset of two vertices for this subtree is $\{5,7\}$ with compression

$$C = (\ell(5) - c)w(5) + (\ell(7) - c)w(7) = 51.$$

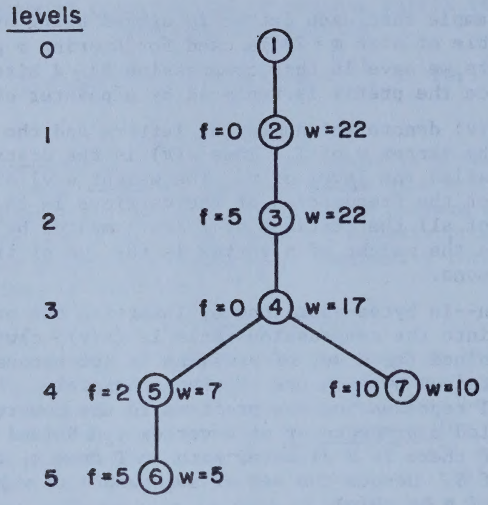


Figure 1.

Note that the compression of the subset {6,7} is only 50.

The subset of three vertices of maximum compression for the entire tree T containing the subset {5,7} for the subtree rooted at 4 is S = {3,5,7} with compression

$$C(S) = (\ell(3)-c)(w(3)-w(5)-w(7)) + (\ell(5)-c)w(5) + (\ell(7)-c)w(7) = 56.$$

However, the optimal subset of three vertices for the entire tree T is $S_1 = \{3,6,7\}$ with compression $C(S_1) = 57$. Thus our straightforward approach may converge to a nonoptimal subset.

In order to understand this phenomenon, we express formula (1) in an alternative form. The subset of S containing no predecessors in S is called the front of S, denoted by F(S). Clearly

$$\sum_{v\varepsilon S} (\ell(v)-c) \sum_{x\varepsilon SC(v)} w(x) = \sum_{v\varepsilon S-F(S)} (\ell(PR(v))-c)w(v).$$

Substituting in (1), we obtain another form for the compression:

$$C(S) = \sum_{v\varepsilon S} (\ell(v)-c)w(v) - \sum_{v\varepsilon S-F(S)} (\ell PR(v))-c)w(v),$$

or

$$C(S) = \sum_{v\varepsilon S-F(S)} (\ell(v) - \ell(PR(v)))w(v) + \sum_{v\varepsilon F(S)} (\ell(v)-c)w(v). \tag{2}$$

Denote by $N(v) = (\ell(v) - \ell(PR(v)))w(v)$ the net compression obtained for a vertex $v\varepsilon S - F(S)$. For convenience we may assume for each vertex $v\varepsilon F(S)$ an artificial predecessor at level c . Then

$$C(S) = \sum_{v\varepsilon S} N(v).$$

This new expression for the compression, in terms of net compression shows that contribution of w(v) to the compression depends on the level of PR(v). Thus it is impossible to find the subset of S which is optimal for the subtree of T rooted at a vertex v without knowing in advance the level of PR(v). However, we can calculate the contribution of v to the compression for each possible level of PR(v). Let $PN(v,i) = (\ell(v)-i)w(v)$ denote the projected net compression obtained for a vertex v, assuming $\ell(PR(v)) = i$.

In the next section we present a dynamic programming procedure for finding a optimal subset, using the projected net compression.

4. AN EFFICIENT DYNAMIC PROGRAMMING PROCEDURE

In this section we present a dynamic programming procedure for finding an optimal subset. First we present a straightforward procedure and later on an improved procedure of polynomial complexity.

We process the vertices of the tree T in end-order. For each vertex v and integers k and i, $1 \leq k \leq m$, $c \leq i \leq \ell(v)$, we compute the maximum projected net compression PNC(v,k,i) obtained for a subset S of k vertices in the subtree rooted at vertex v, under the assumption that the first vertex in S on the path from v to the root of T is at level i. In case no vertex S is on this path, we assume existence of an artificial vertex on level c in order to handle the vertices in the front of S similarly to the other vertices. (Note that it is not economical to have a vertex of S at any level $i < c$.)

Suppose a vertex v has d sons $x_1, x_2, \cdots, x_d$ in T, for whose subtrees the required information has already been computed, and we want to compute the projected net compression PNC(v,k,i) for $1 \leq k \leq m$ and $c \leq i \leq \ell(v)$. Define $PNC(v,0,i) = 0$ for $c \leq i \leq \ell(v)$. Also for every vertex v having less than k-1 descendants, $PNC(v,k,i) = 0$. Let $\pi(k,d)$ denote the set of all ordered partitions of k elements into d subsets. We use the terms PC1(v,k,i) and PC2(v,k,i) to handle separately the following two cases.

Case 1: $v \in S$. Let $(n_1, n_2, \cdots, n_d) \in \pi(k-1,d)$. Define

$$PC1(v,k,i) = \max_{\pi(k-1,d)} \sum_{j=1}^{d} PNC(x_j, n_j, \ell(v)) + (\ell(v)-i)w(v).$$

Case 2: $v \notin S$. Define

$$PC2(v,k,i) = \max_{\pi(k,d)} \sum_{j=1}^{d} PNC(x_j, n_j, i).$$

Now $\qquad PNC(v,k,i) = \max(PC1(v,k,i), PC2(v,k,i)).$

Then the maximum compression obtained for the optimal subset S is $C(S) = PC2(r,m,c)$, where r is the root of T.

It is easy to show that this procedure satisfies the principle of optimality (see e.g. Dreyfus and Law [2]). Hence the procedure is valid.

By keeping appropriate pointers while computing the values in the tables, the optimal subset itself can easily be reconstructed.

Let us demonstrate this procedure for an optimal subset of three vertices for the tree T of Figure 1, assuming $c = 1$. The values of PC1(v,k,i), PC2(v,k,i) and PNC(v,k,i) appear in Tables 1, 2 and 3 respectively. An italicized number in the tables PC1 and PC2 indicates that this value is chosen to the table of PNC-values. The optimal subset of 3 vertices realizing the compression of 57 is $S = \{3,6,7\}$. Note that changing f(3) from 5 to 3 will yield the optimal subset {5,6,7} with compression of 56.

The problematic point in the complexity analysis of the above procedure is the number of partitions involved. For example, the number of ordered partitions of m into d parts is $\binom{m+d-1}{d-1} = O(m^{d-1})$ (see Hall [4]). Thus the complexity of the procedure might be quite high.

In order to decrease the complexity of the procedure we replace the straightforward computation of PC1(v,k,i) and PC2(v,k,i) by a step-by-step

(v,k) / i	(7,1)	(6,1)	(5,1)	(5,2)	(4,1)	(4,2)	(4,3)	(3,1)	(3,2)	(3,3)	(2,1)	(2,2)	(2,3)
1	$\underline{30}$	$\underline{20}$	$\underline{21}$	$\underline{26}$	$\underline{34}$	44	54	22	42	$\underline{57}$	-	34	51
2	$\underline{20}$	$\underline{15}$	14	$\underline{19}$	17	27	37	-	-	-	-	-	-
3	$\underline{10}$	$\underline{10}$	7	$\underline{12}$	-	-	-	-	-	-	-	-	-
4	-	$\underline{5}$	-	-	-	-	-	-	-	-	-	-	-

Table 1. PC1(v,k,i)

(v,k) / i	(7,1)	(6,1)	(5,1)	(5,2)	(4,1)	(4,2)	(4,3)	(3,1)	(3,2)	(3,3)	(2,1)	(2,2)	(2,3)	(1,3)
1	-	-	20	-	30	$\underline{51}$	$\underline{56}$	$\underline{34}$	$\underline{51}$	56	$\underline{34}$	$\underline{51}$	$\underline{57}$	57
2	-	-	$\underline{15}$	-	$\underline{20}$	$\underline{35}$	$\underline{39}$	-	-	-	-	-	-	-
3	-	-	$\underline{10}$	-	-	-	-	-	-	-	-	-	-	-
4	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 2. PC2(v,k,i)

(v,k) / i	(7,1)	(6,1)	(5,1)	(5,2)	(4,1)	(4,2)	(4,3)	(3,1)	(3,2)	(3,3)	(2,1)	(2,2)	(2,3)
1	30	20	21	26	34	51	56	34	51	57	34	51	57
2	20	15	15	19	20	35	39	-	-	-	-	-	-
3	10	10	10	12	-	-	-	-	-	-	-	-	-
4	-	5	-	-	-	-	-	-	-	-	-	-	-

Table 3. PNC(v,k,i)

process. Suppose that a vertex v has d sons $x_1, x_2, \cdots, x_d$. Let $T_v(j), 1 \leq j \leq d$, denote the subtree of T rooted at v, containing the sons $x_1, x_2, \cdots, x_j$ of v and all their descendants. Let $PC1(v,k,i,j)$ and $PC2(v,k,i,j)$ denote the values of $PC1(v,k,i)$ and $PC2(v,k,i)$ for the subtree $T_v(j)$, respectively. Note that $PC1(v,k,i) = PC1(v,k,i,d)$ and $PC2(v,k,i) = PC2(v,k,i,d)$. Define $PC1(v,0,i,j) = PC2(v,0,i,j) = 0$ for $c \leq i \leq \ell(v)$, $1 \leq j \leq d$. Also for every vertex v with less than k-1 descendants, $PC1(v,k,i,j) = PC2(v,k,i,j) = 0$. Using these new terms we obtain now the following dynamic programming procedure.

Again we process the vertices of the tree in endorder. For each vertex v with sons $x_1, x_2, \cdots, x_d$ and integers k and i, $1 \leq k \leq m$, $c \leq i \leq \ell(v)$, we perform the following computation:

$$PC1(v,k,i,1) = PNC(x_1,k-1,\ell(v)) + (\ell(v)-i)w(v)$$

$$PC2(v,k,i,1) = PNC(x_1,k,i)$$

and for $j = 2,3,\cdots,d$,

$$PC1(v,k,i,j) = \max_{1 \leq q \leq k} (PC1(v,q,i,j-1) + PNC(x_j,k-q,\ell(v)))$$

$$PC2(v,k,i,j) = \max_{0 \leq q \leq k} (PC2(v,q,i,j-1) + PNC(x_j,k-q,i))$$

and

$$PNC(v,k,i) = \operatorname{Max}(PC1(v,k,i), PC2(v,k,i)) .$$

The compression for an optimal subset is $C(S) = PC2(r,m,c)$, where r is the root of T. The validity of this step-by-step procedure is easily derived from the validity of the above straightforward procedure.

Consider now the complexity of this procedure. The computation of $PNC(v,k,i)$ for a vertex v with d(v) sons requires $O(k \cdot d(v))$ operations. Thus the total number of operations for the vertex v is

$$\sum_{k=1}^{m} \sum_{i=c}^{\ell(v)} O(k \cdot d(v)) = O(m^2 \ell(v) d(v)) .$$

Let $|T|$ denote the number of vertices in T and let h(T) denote the height of T. Now $\sum_{v \varepsilon T} d(v) = |T| - 1$ since T is a tree, and $\ell(v) < h(T)$. Hence the complexity of this dynamic programming procedure is

$$\sum_{v \varepsilon T} O(m^2 \ell(v) d(v)) = O(m^2 |T| \cdot h(T)).$$

Note that h(T) is the number of letters in the longest given word and $|T|$ is bounded by the sum of the lengths of the given words which are not a prefix of another given word.

The given procedure is also efficient with regard to storage requirements, since only two values for two consecutive j's are simultaneously required for each triple (v,k,i) in the tables. Furthermore, using the endorder processing of a tree, it is sufficient to keep the table of one vertex for each level of the tree in addition to the currently processed vertex. Hence the total storage required is bounded by $O(m \cdot h^2(T))$.

REFERENCES

1. V. L. Doucette, K. M. Harrison and E. J. Schuegraf, A comparative evaluation of fragment dictionaries for the compression of French, English and German bibliographic data bases, In: Proc. 3rd Intern. Conference on Computing in the Humanities (S. Lusignan and J. S. North, eds.), University of Waterloo Press, Waterloo, pp. 297-305, 1977.
2. S. E. Dreyfus and A. M. Law, The Art and Theory of Dynamic Programming, Academic Press, New York, 1977.
3. D. Gotlieb, S. A. Hagerth, P. G. H. Lehot and H. S. Rabinovitz, A classification of compression methods and their usefulness for a large data processing center, National Comp. Conference 44 (1975), 453-458.
4. M. Hall, Combinatorial Theory, Blaisdell, Waltham, MA, 1967.
5. T. Radhakrishnan, Selection of prefix and postfix word fragments for data compression, Inform. Process. & Management 14 (1978), 97-106.
6. F. Rubin, Experiments in text file compression, Comm. ACM 19 (1976), 617-623.

DECENTRALIZED LOAD FLOW ALGORITHM FOR LARGE-SCALE POWER SYSTEM

HUNGWEN LI
RCA Advanced Technology Laboratories
Camden, NJ

MARLIN H. MICKLE and WILLIAM G. VOGT
Department of Electrical Engineering, University of Pittsburgh
Pittsburgh, PA

ABSTRACT

This paper concerns the development of a decentralized load flow algorithm for a large-scale power system. Decentralization was achieved by "physically" decomposing the power network such that the data for subnetworks can be collected locally and as a result, a centralized database — hence, a large amount of data transferring — can be avoided.

Executed sequentially, the algorithm is a block Gauss-Seidel method. However, with the minimized cut-nodes due to the physical network decomposition, the algorithm is faster than that of logical decomposition.[1] On its parallel form, the algorithm achieves speedup greater than 3*N where N is the number of processors in this study.

INTRODUCTION

The load flow is the fundamental computation to the power system control. As systems grow, the load flow computation burden becomes the bottleneck of the realtime control for large-scale power systems. An optimal ordering algorithm has been proposed[1] to speed up the computing. The algorithm assumes a centralized database and results in a "logical" decomposition which is, in general, not naturally corresponding to the "physical" network decomposition, and therefore, cannot be implemented in a "decentralized" fashion without transferring large amounts of data.

This paper investigated empirically the performance of a decentralized load flow algorithm based on "physical" network decomposition. Decentralization is achieved in two senses: (1) the computation uses only local data (except the incident nodes) and hence it eliminates the necessity of a centralized database; and (2) the adopted decomposition algorithm[2] minimizes the number of boundary nodes and hence yields very loosely-coupled subnetworks.

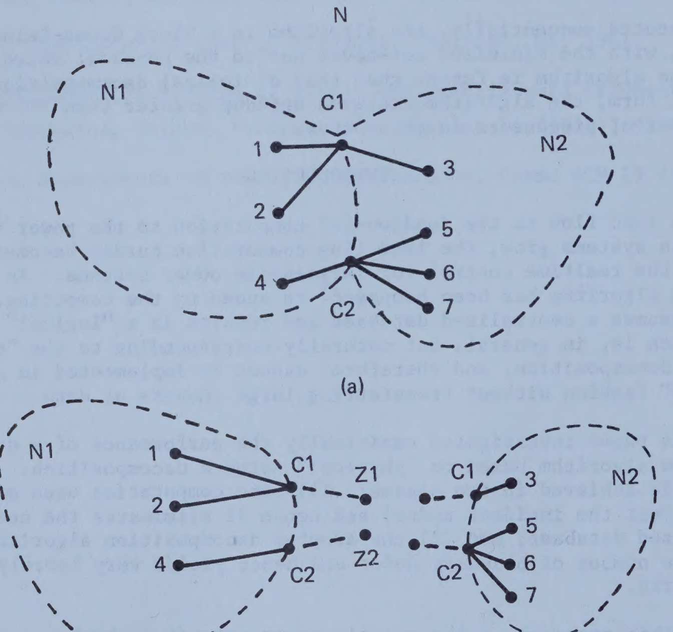
In this approach, a large-scale power network is broken into subnetworks of smaller size. These subnetworks are then solved independently by the existing method except for the consideration of the interrelationship between subnetworks. The collection of solutions of the subnetworks (corrected for interactions) is the solution of the original large-scale network.

This network decomposition approach is further enhanced by the concept of parallel processing to overcome the limitation of using a single processor. The solving of each subnetwork can be handled in one processor and all subnetwork solutions can be done simultaneously at the expense of the number of processors. This involves the upgrading of the existing solution method in the multiple-processor environment, which makes the industry load flow code directly applicable to a multiple-microcomputer system.

The algorithm was implemented[3] based on the revised version of an industry load flow code and was tested using a 442-node power network decomposed into 2,4, and 8 subnetworks.

PHYSICAL NETWORK DECOMPOSITION

A power network is usually represented by a network, N, with nodes representing the generator stations or the load centers, and with branches representing the transmissions lines. This network N can be decomposed into smaller subnetworks, for example N_1 and N_2, through a collection of cut-nodes, C_1 and C_2, as illustrated in Fig. 1(a). Two functions related to the nodes



$$T_{o1}(C1) = T_{s2}(C1) = \{3\}$$
$$T_{o1}(C2) = T_{s2}(C2) = \{5, 6, 7\}$$
$$T_{s1}(C1) = T_{o2}(C1) = \{1, 2\}$$
$$T_{s1}(C2) = T_{o2}(C2) = \{4\}$$

(b)

Fig. 1. Network decomposition, subnetwork interrelationship, and definition of T_o and T_s.

incident to the cut-node n, $T_{sk}(n)$ and $T_{ok}(n)$, are defined for subnetwork N_k as

$T_{ok}(n) = \{$nodes that are incident to cut-node n and do not belong to $N_k\}$;

$T_{sk}(n) = \{$nodes that are incident to cut-node n and belong to $N_k\}$.

The subscript o is for "other" and s is for "self". Examination of Fig. 1(b) reveals the meaning of these definitions.

Consider a network N consisting of k subnetworks, N1, N2,...,Nk, that are interrelated through a set of cut-nodes $[C=C_i]$. These cut-nodes can be viewed as having the property that they are the only entries through which the electrical effect from the neighboring subnetworks inject. The electrical effect is denoted by a pair (E_{zij}, Y_{zij}) for each cut-node C_j in the subnetwork N_i, where E_z is the voltage of the auxiliary node z and Y_z is the admittance of the auxiliary branch b shown in Fig. 1(b).

By equivalencing the power flow and the admittance before and after the decomposition for each cut-node, we obtain the expression for E_{zij} and Y_{zij} as follows:

$$g_{zij} = \sum_{k \epsilon T_{oi}(j)} g_{jk}$$

$$b_{zij} = \sum_{k \epsilon T_{oi}(j)} b_{jk} \tag{1}$$

and

$$e_{zij} = \frac{(\Sigma g_{jk} e_k + \Sigma b_{jk} f_k)(\Sigma g_{jk}) + (\Sigma b_{jk} e_k + \Sigma g_{jk} f_k)(\Sigma b_{jk})}{(\Sigma g_{jk})^2 + (\Sigma b_{jk})^2}$$

$$f_{zij} = \frac{(-\Sigma g_{jk} e_k + \Sigma b_{jk} f_k)(\Sigma b_{jk}) + (\Sigma b_{jk} e_k + \Sigma g_{jk} f_k)(\Sigma g_{jk})}{(\Sigma g_{jk})^2 + (\Sigma b_{jk})^2} \tag{2}$$

for any cut-node j in the subnetwork, and all summations are summed over $k \epsilon T_{oi}(j)$, where $Y_{zij} = g_{zij} + jb_{zij}$ and $E_{zij} = e_{zij} + jf_{zij}$. Detailed derivation of equations 1 and 2 can be found in Lee.[2]

So treated, the load flow problem associated with each N_i can be solved independently and, at the same time, can consider the effect of other subnetworks. The overall solution will be collected from that of individual subnetworks. Most importantly, the approach of "physical" network decomposition collects data for each subnetwork locally, and as a result, the database for the algorithm is decentralized and a large amount of data transferring can be avoided.

The operating environment for the parallel Newton-Raphson (N-R) method is described before presenting the parallel N-R method. There are k processors, Mp1, Mp2, ..., Mpk, with each charged for the N-R solution for one subnetwork. Each processor is equipped with one STOP/GO register indicating whether the processor should stop or go. These registers are located in the shared-memory such that they are accessible by all Mps. One of the processors, SMP, will be additionally assigned the responsibility of checking the convergence criteria and controlling the STOP/GO conditions of all the processors. With this kind of environment, we are ready to address the parallel N-R method.

We designate δ as phase angle of node voltage;
$|E|$ as magnitude of node voltage;
P as real power;
and Q as imaginary power.

For subnetwork N_i having n_i nodes, t_i cut-nodes and z_i auxiliary nodes, we define

$$x_i = [\delta_{i1}\ \delta_{i2} \cdots \delta_{in-1}\ |E_{i1}|\ |E_{i2}| \cdots |E_{ini}|]^T \tag{3}$$

$$\Delta x_i = [\Delta\delta_{i1}\ \Delta\delta_{i2} \cdots \Delta\delta_{in-1} \Delta\,|E_{i1}|\,\Delta\,|E_{i2}| \cdots \Delta\ |E_{ini}|]^T \tag{4}$$

$$F_i(X_i,\ \overline{E}_{zi}) = [\Delta P_{i1}\ \Delta P_{i2} \cdots \Delta\ P_{ini}\ \Delta Q_{i1}\ \Delta Q_{i2} \cdots \Delta\ Q_{ini}]^T \tag{5}$$

where

$$P_{ij} = P_{ij}\ \text{(schedule)} - P_{ij}\ \text{(calculated)}$$

$$= P_{ij}\ \text{(schedule)} - \text{Re}\left[|E_{ij}|e^{-j\delta_{ij}} \sum_{m=1}^{n_i} Y_{jm}|E_{im}|e^{j\delta_{im}} + \sum_{m=1}^{z_i} Y_{jm}|E_{im}|e^{j\delta_{im}}\right]$$

$$= P_{ij}\ \text{(schedule)} - \text{Re}\left[|E_{ij}|e^{-j\delta_{ij}} \sum_{m=1}^{n_i} Y_{jm}|E_{im}|e^{-j\delta_{im}}\right] - \text{Re}\left[|E_{ij}|\ e^{-j\delta_{ij}} \sum_{m=1}^{z_i} Y_{jm}|E_{im}|e^{-j\delta_{im}}\right] \tag{6}$$

and

$$Q_{ij} = Q_{ij}\text{ (schedule) } - \text{Im}\left[|E_{ij}| e^{-j\delta_{ij}} \sum_{m=1}^{n_i} Y_{jm} |E_{im}| e^{-j\delta_{im}} \right]$$

$$\text{Im}\left[|E_{ij}| e^{-j\delta_{ij}} \sum_{m=1}^{z_i} Y_{jm} |E_{im}| e^{-j\delta_{im}} \right] \qquad (7)$$

The Jacobian matrix for the Ni is defined as

$J_i\ (X_i, \overline{E}_{zi})_{rs} = \partial F_{ir} / \partial X_{is}$.

The parallel N-R method operating in the multiple microcomputer system is stated as follows.

For MPi, do (S1) to (S5)

(S1) Sets initial values for X°_{il}, l=1, n_i;

Stores X°_{ij} into shared-memory, $j \epsilon T_{si}$ (1), i=1, t_i;

(S2) Computes $\overline{Y}_{zij}$, j=1, t_i according to equation 1;

For (r+1)th iteration, do (S3) to (S5)

(S3) Fetches X^r_{mj} from the shared-memory, $j \epsilon T_{oi}$ (1) i=1, t_i where m lies between 1 to k and m ≠ i;

(S4) Computes $\overline{E}^r_{zij}$, j=1, t_i, according to equation 2;

(S5) Forms $J_i\ (X^r_i, E^r_{zi})$ and $F_i\ (X^r_i, \overline{E}^r_{zi})$;

Solves $F_i\ (X^r_i, \overline{E}^r_{zi}) = -J_i\ (X^r_i, \overline{E}^r_{zi}) \cdot \Delta X^r_i$ for ΔX^r_i;

Obtains $X^{r+1}_{ij} = X^r_i + \Delta X^r_i$;

Stores X^{r+1}_{ij} in shared-memory, $j \epsilon T_{si}$ (1), j = 1, t_i;

Resets STOP/GO register to STOP condition and waits for GO condition to continue the iterative procedure starting from (S3).

For SMP, do (S6)

(S6) Checks covergence criteria;
Terminates the iterative procedure, if allowable; otherwise, sets all STOP/GO registers to GO condition.

The parallel N-R method was implemented as a FORTRAN program, PARALF[3], on a PDP-10 system at the University of Pittsburgh. It is based on the revised version[4] of an industry load flow program produced by the System Planning Division of the Philadelphia Electric Company.[1,5] The heart of PARALF consists of three subroutines: MATGEN to formulate the Jacobian matrix; ELIM to perform the sparse matrix triangular factorization; and BAKSUB to perform the backward substitution to obtain X.

Subroutines ELIM and BAKSUB are adopted from the standard load flow code without any change of the FORTRAN statements. In MATGEN, two statements are added to the subroutine. Also the other subroutines are 100% preserved. Our approach to implement the Parallel N-R method is to add two subroutines into the industry load flow program to compute the admittance of the auxiliary nodes (S2) and the voltage of the auxiliary nodes (S4), plus the coordination between subnetworks. Obviously, the augmentation is straightforward and easy to implement compared with the original industry load flow program. By so doing, we can save substantial engineering effort and can achieve an immediate benefit. The concept of upgrading the existing method to the parallel processing level proposed in this paper makes the industry load flow code directly applicable to a multiple-microcomputer system.

RESULT

The proposed algorithm can be executed in either a single-processor or multiprocessor environment, both of which were tested under a real power network. A 442-node power network was decomposed into 2, 4, and 8 subnetworks and the load flow algorithm was executed on 1, 2, 4, and 8 processors (where applicable). For the single-processor case, about a 4:1 speedup can be achieved due to the reduced fill-in and simpler equivalencing (equations 1 and 2). For the multiprocessor case, roughly 3*N:1 (where N is the number of processors utilized) speedup was obtained because of the combined effect of parallel processing and the "physical" network decomposition.

Executed sequentially, the algorithm is a block Gauss-Seidel method; however, the overhead represented by (S3) and (S4) is much less than the solving of the bordered block-diagonal form resulted from the "logical" decomposition approach.[1] Figure 2 illustrates the performance of the algorithm in the single-processor environment. It is also indicated in Fig. 1 that the overhead increases as the number of subnetworks increases; hence, the optimal subnetwork number is a function of the ratio of (S5) to the overhead.

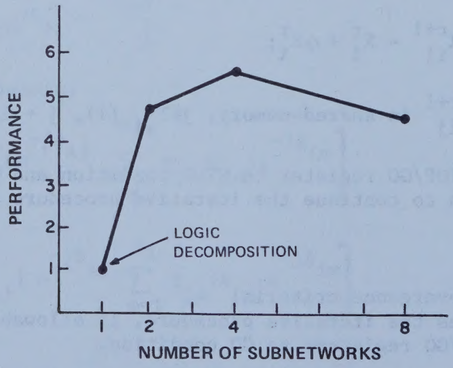


Fig. 2. Performance of sequential execution (1-processor) of the algorithm.

Executed in the multiprocessor environment, the algorithm combines the effect of parallel processing and the "physical" network decomposition; thus it achieves a speedup greater than 3*N (where N is the number of processors) in this study as shown in Fig. 3. Considering the performance improvement solely due to the parallel processing, an average of 80% of the processor's power is utilized. The difficulty to approach 100% utilization is due to the unbalanced decomposition, the larger percentage of the overhead when subnetwork size becomes smaller, and the slower convergence rate.

Compared with other similar studies,[6,7] our approach decomposes the network through nodes rather than branches and our equivalence equation is much simpler and hence incurres less overhead. Due to the inexact equivalencing rather than exact,[6,7] the convergent characteristic is different from the one-piece solution; however, no major difficulty was encountered in finding a decomposition that converges at the same rate. Most interestingly, the solution from this algorithm is identical to that of the one-piece algorithm up to the third digit beyond the decimal point.

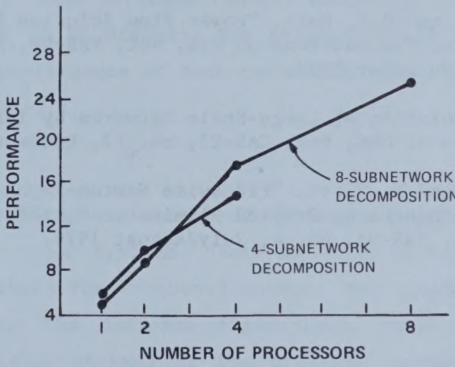


Fig. 3. Performance of parallel execution of the algorithm.

CONCLUSION

This paper has developed a decentralized load flow algorithm which avoids the necessity of a central database and establishes an applicable way of upgrading the existing load flow code to the parallel processing environment.

The generalization of this study is a methodology which combines the concept of decomposition and parallel processing such that it can fully utilize the existing method and at the same time, achieve parallel processing among the decomposed subsystems. It is evident that the methodology is of great value for large-scale engineering problems because first the problem can be solved faster and cheaper, second the effect accumulated in the existing method is not wasted, and third the curse of the ever increasing dimensionality is reduced. The limitation of this methodology lies only in the imagination of the interpretation of the relationship among the subsystems.

REFERENCES

1. W.F. Tinney and J.W. Walker, "Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization", IEEE Proceedings, vol. 55, pp. 1801-1809, November 1967.
2. J.G. Lee, "Optimal Decomposition of Large-Scale Network — Its Application to Power Flow Problems", unpublished doctoral dissertation, Department of Electrical Engineering, University of Pittsburgh, March 1977.
3. H. Li, "Load Flow Program PARALF and NOLF11 User's Guide and Program List", Technical Report No. 79-001, Department of Electrical Engineering, University of Pittsburgh, December 1979.
4. J.G. Lee, "Power Flow Program PFNR for User's Guide and Program List", Technical Report No. LSSG-TR-77-103, Department of Electrical Engineering, University of Pittsburgh, March 1977.
5. W.F. Tinney and C.E. Hatt, "Power Flow Solution by Newton's Method", IEEE Transactions on PAS, vol. PAS-86, no. 11, pp. 1449-1460, November 1967.
6. F.F. Wu, "Solution of Large-Scale Networks by Tearing", IEEE Transactions on CAS, vol. CAS-23, no. 12, December 1976.
7. R. Kasturi and M. Potti, "Piecewise Newton-Raphson Load Flow — An Exact Method Using Ordered Elimination", IEEE Transactions on PAS, vol. PAS-95, no. 4, July/August 1976.

REDUCTION OF HIGH ORDER SYSTEM WITH APPLICATION TO CONTROLLER DESIGN

I. EL-NAHAS, N.K. SINHA and R.T.H. ALDEN

Group on Simulation, Optimization and Control
Faculty of Engineering
McMaster University
Hamilton, Ontario, Canada L8S 4L7

Abstract:

In this paper, seven different methods for model reduction have been used for obtaining third-order models of an eleventh-order system, representing a synchronous machine connected to an infinite bus-bar through a transmission line. Each of these reduced models is utilized for designing PID controllers so as to minimize the integral-square error as a performance index. The performance of each controller with the original system is examined and compared.

INTRODUCTION

Over the past decade, a number of different methods have been proposed for reducing high order systems. Some of these methods are developed for time domain and others for frequency domain. One popular approach has been based on retaining the dominant eigenvalues (small eigenvalues) or the dominant states (slow states) of the original system [3,5,12]. Another approach has been based on fitting a low-order model such that its time response to a specified input, or its frequency response is an optimum approximation to the corresponding response of the original system [1,6,10,16]. Other methods have been based on matching some time moments or Markov parameters of the reduced models to those of the original system [4,7,8,14,15]. All of these methods attempt to obtain a low-order model which approximates, as far as possible, certain characteristics of the original system.

It is obviously desirable that the reduction should be computationally simple and applicable to a wide class of systems. Furthermore, it should yield a stable reduced order - model for a given stable large system. As a reduced order model cannot characterize a given system as completely as the large order model, it would be desirable to determine how closely and in what sense a particular aspect of the behaviour of a given large system has been approximated by a particular reduced model. But the above criteria for comparison should not be the only goal, more important is the application of model reduction to the design of a feedback control system. The object of this paper is to examine this aspect for several different methods of model reduction when the low-order models are utilized for designing PID controllers so as to minimize the integral square error as a performance index. The comparison is made for an 11th - order system representing a synchronous machine connected to an infinite bus-bar through a transmission line. Seven methods are utilized for obtaining 3rd order models. These are the method of (i) aggregation, (ii) aggregation with moment matching, (iii) continued fraction expansion, (iv) Routh approximation, (v) Routh approximation with moment matching, (vi) singular perturbation and (vii) frequency domain approximation (Elliott and Wolovich 1980). These methods have been chosen as they are representative of the different reduction approaches. A controller is designed for each case, and the performance with the original system is examined. This provides a very interesting comparison and gives good insight into the application of these methods for such design.

METHODS OF REDUCTION:

The existing model reduction techniques can be classified into three main groups. The first group of methdos preserve either the dominant eigenvalues or the dominant states of the original system. The methods of Davison (1966), and Aoki (1968) belong to this category as they retain the dominant eigenvalues. Besides having small negative real parts, these modes must be controllable and observable. Another method which retains the slowly varying states of the system by neglecting the fast phenomena and improving the approximation by reintroducing their effect as 'boundary layer' corrections is the singular perturbation method (Sannuti and Kokotvic 1968).

The second group of methods involves fitting a low-order model such that its time response to a specified input, or its frequency response is an optimum approximation to the corresponding response of the original system. Some of such methods are those due to Anderson (1967), Sinha and Pille (1971) which approximate time responses and that due to Elliott and Wolovich (1980) which matches the frequency response at certain prescribed frequencies.

The third group of methods is based on matching some time moments or parameters of the reduced models to those of the original system. The continued fraction expansion method of Chen and Shieh (1968) was the earliest method of this type, and was later shown to be a special case of Padé approximation (Shamash 1974). An important drawback of such methods is that the low-order model may turn out to be unstable even if the original system is stable. The development of the Routh approximation method (Hutton and Friedland 1975) overcame the problem described above. Another method of overcoming this difficulty is to combine aggregation with matching time moments (Hickin and Sinha 1980). Also, combining Routh method with matching time moments (Sinha, El-Nahas and Alden 1980) overcomes this problem as well as retains the advantage of matching time moments (and or Markov parameters). In addition, it provides more flexibility in choosing the degree of the numerator.

DESIGN PROCEDURE

The problem is to design a PID conteroller acting on the error $e(t) = y_r(t) - y(t)$ so that the integral-square error is minimized, where $y_r(t)$ is the reference output and $y(t)$ is the actual output. The PID controller has the following form

$$u(t) = K_p\, e(t) + K_I \int e(t)\, dt + K_d\, \dot{e}(t) \tag{1}$$

The design procedure involves four steps. The first step is to express the error as a function of the complex frequency s. This function will involve the parameters of the controller as unknown coefficients. The second step is to express the integral-square error I in terms of the error transform E(s) by Parseval's theorem; i.e.,

$$I = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} E(s)\, E(-s)\, ds \tag{2}$$

At this stage, provided E(s) is a rational function, the integral square error will appear in the form

$$I = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} \frac{b(s)\, b(-s)}{a(s)\, a(s)}\, ds \tag{3}$$

where b(s) and a(s) are polynomials in s. The third step is to evaluate the integral. Fortunately, definite integrals of this form have been evaluated in terms of the coefficients appearing in the polynomials [11]. The fourth step is to adjust the values of the controller parameter in such a way as to minimize the integral square error.

EXAMPLE:

The example considered for the study is the model of a synchronous machine connected to an infinite bus-bar through a transmission line. The linearized state equations are obtained using the P.Q.R. method (Anderson 1969) in the form.

$$\dot{\underline{x}} = A\,\underline{x} + b\,u \quad , \quad y = cx \tag{3}$$

where $\underline{x}^T = [\Delta\delta,\ \Delta w,\ \Delta\psi_{fd},\ \Delta\psi_d,\ \Delta\psi_{kd},\ \Delta\psi_q,\ \Delta\psi_{kq},\ \Delta p_m,\ \Delta g,\ \Delta v_v,\ \Delta v_f]$

$u = v_{ref}$

$y = \Delta v_t$.

The model is of 11th order, and has the following eigenvalues.

$-10.3 \pm j\ 572$, $-\ 503$, -89.9, $-.48 \pm j\ 11.6$, $-15.2 \pm j\ 11.6$, -11.0, -4.45 and -0.909

Third-order models of this system were obtained using the various methods mentioned in the previous section. These are given below

(1) Aggregation

$$G_1(s) = \frac{6.625\ s^2 + 8238.57\ s + 2059315.0}{(s + .909)(s^2 + 30.4\ s + 365.0)}$$

(2) Aggregated model matching T_1 and T_2

$$G_2(s) = \frac{1595.47\ s + 175768.5}{(s + 0.909)(s^2 + 30.4\ s + 365.0)}$$

(3) Singular perturbation

$$G_3(s) = \frac{16.76\ s^2 + 50.16\ s + 1308}{(s + 15.359)(s^2 + 3.915\ s + 60.832)}$$

(4) Continued fraction expansion

$$G_4(s) = \frac{7.82\ s^2 + 23.9\ s + 16.96}{(s + 0.82)(s^2 + 7.98\ s + 18.6)}$$

(5) Routh approximation

$$G_5(s) = \frac{8.56\ s^2 + 23.34\ s + 17.55}{(s + 0.735)(s^2 + 8.64\ s + 21.578)}$$

(6) Routh approximation model matching T_1 and T_2

$$G_6(s) = \frac{7.287\ s^2 + 25.2\ s + 15.15}{(s + 0.735)(s^2 + 8.64\ s + 21.538)}$$

(7) Method of Elliott and Wolovich

$$G_7(s) = \frac{14.8\ s^2 + 209.1\ s + 38.2}{(s + 0.36)(s + 10.86)(s + 17.58)}$$

The responses of the 11th. order system and each of the seven reduced models to a unit step are shown in fig. (1) and (2). It is seen that the models obtained by aggregation and singular perturbation give a noticeable error at steady state, while the other models introduce small deviations. Since the fast transients have been neglected in the reduced models, all the models are giving different responses at the initial part. The response due to the aggregation with moment matching model is a little closer to the response of the original system for this part.

Now for the original system and each of the reduced models a PID controller is designed to minimize the integral-square error, with the integral square of the input derivative constraint to be 5.0. The controllers, as well as the cost function which are obtained, are respectively:

$$g_0 = 13.5 \quad (1 + \frac{1}{2.06s} + 1.52\ s) \qquad , I_0 = 0.0661$$

$$g_1 = 0.095 \quad (1 + \frac{1}{0.148s} + 0.0371\ s) \qquad , I_1 = 0.79524$$

$$g_2 = 0.056 \quad (1 + \frac{1}{0.259s} + 0.197\ s) \qquad , I_2 = 0.6338$$

$$g_3 = 0.0446 \quad (1 + \frac{1}{1.9\ s} + 0.314\ s) \qquad , I_3 = 2.2065$$

$$g_4 = 2.181 \quad (1 + \frac{1}{1.38s} + 1.34\ s) \qquad , I_4 = 0.1132$$

$$g_5 = 2.09 \quad (1 + \frac{1}{2.41s} + 0.77\ s) \qquad , I_5 = 0.2126$$

$$g_6 = 2.0 \quad (1 + \frac{1}{1.48s} + 1.315\ s) \qquad , I_6 = .1402$$

$$g_7 = 0.015 \quad (1 + \frac{1}{1.8s} + 1.225\ s) \qquad , I_7 = 6.38$$

The subscript 0 is used for the controller for the original system, while subscripts 1 to 7 are used for the corresponding reduced models. It should be noted that the controllers are quite different. The closed loop responses of the controllers with the corresponding reduced models to unit step input are calculated and compared with the closed loop performance of each controller with the original system to the same unit step input.

These responses are shown in Figs. (3), (4), (5) and (6). It is seen that the responses of the various models (original as well as reduced-order models) with the corresponding controller are quite close to each other except the one obtained by the method of Elliott and Wolovich which gives a noticeable overshoot. Also, there are small oscillations shown in the response of the original system. But, examining the responses of each controller with the original system it is seen that they are different from the response of the original system, g_1, g_2, g_4, g_5 and g_6 give different responses in the initial part. They make the system slightly damped while g_3 and g_7 introduce relatively high overshoot. Also, it is seen that the controller g_0 with the original system gives the lowest value of the cost function while g_3 and g_7 give the largest values.

CONCLUSIONS

Several existing model reduction techniques have been reviewed and compared for designing PID controllers so as to minimize the integral square error. It is seen that the responses of the various models with the corresponding controller are quite close, while the responses due to each controller with the original system are noticeably different in the initial part from the response of the original system with its own controller. This is probably due to the fact that the faster modes have been neglected. Also, it is seen that all the responses are approximately the same at steady state.

Thus, if we were concerned only with the steady-state response of the closed-loop system, all the reduction methods would appear equally good. However, the controller designed with the original system gives the lowest value of the cost function, other controllers based on reduced-order models result in larger values for this cost function. Among the various controllers from the reduced model, the best results were obtained with the methods using the continued fraction expansion and the Routh approximation method combined with moment matching. This would appear reasonable since the time responses for these cases were closer to that of the original system.

REFERENCES

1. Anderson, J.H. (1969), "Geometrical Approach to Reduction of Dynamical Systems", Proc. IEE, vol. 114, pp. 1014-1018.

2. Anderson, J.H. (1969), "Matrix Method for the study of A Regulated Synchronous Machine", IEEE Proc., Vol. 57, pp. 2122-2136.

3. Aoki, M. (1968), "Control of Large Scale Dynamic Systems by Aggregation", IEEE Trans. AC-13, pp. 246-253.

4. Chen, C.F. and Shieh, L.S. (1968), "A Novel Approach to Linear Model Simplification", Int. J. Control, vol. 8, No. 6, pp. 561-570.

5. Davison, E.J. (1966), "A Method for Simplifying Linear Dynamic Systems", IEEE Trans. AC-11, pp. 93-101.

6. Elliott, H. and Wolovich, W.A., (1980), "A Frequency Domain Model Reduction Procedure", Automatica, vol. 16, pp. 167-178.

7. Hickin, J. and Sinha, N.K. (1980), "Model Reduction For Linear Multivariable Systems", IEEE Trans. AC-25, pp. 1121-1127.

8. Hutton, M.F. and Friedland, B., (1975), "Routh Approximation for Reducing Order of Linear Time-Invariant Systems", IEEE Trans. AC-20, pp. 329-337.

9. Kuppurajulu, A. and Elanjovan, S. (1971), "Simplified Power system models for Dynamic Stability Studies", IEEE Trans, vol. PAS-90, No.1, pp. 11-23.

10. Langholz, C. and Bistritz, Y. (1978), "Model Reduction of Dynamic Systems Over a Frequency Interval: 16th. Ann. Allerton Conf. Communication, Control and Computing, pp. 903-912.

11. Newton, G., Gould, L. and Kaiser, J. (1957), "Analytical Design of Linear Feedback Controls" John Wiley & Sons, Inc.

12. Sannuti, P. and Kokotvic, P.V. (1969), "Near Optimum Design of Linear Systems Using Singular Perturbation Method", IEEE Trans., vol. AC-14, pp.15-21.

13. Seraji, H. (1980), "Design of Multivariable PID Controllers for Pole Placement", Int. J. Control, vol. 32, No.4, pp. 661-668.

14. Shamash, Y. (1974), "Stable Reduced-Order Models Using Padé-type Approximations", IEEE Trans., vol. AC-19, P.615.

15. Shieh, L.S., Wer, Y.J., Lue, H.C., Yates, R. and Leonard (1976), "Two Methods for Simplifying Multivariable Systems with various Number of Inputs and outputs", Int. J. System Sci., vol. 7, No. 5, pp. 501-512.

16. Sinha, N.K. and Pille, W. (1971), "A New Method for Reduction of Dynamic system", Int. J. Control, vol. 14, pp. 111-118.

17. Sinha, N.K., El-Nahas, I. and Alden, R.T.H. (1980), "Routh Approximation of Multivariable Systems", 18th Ann. Alerton Conf. on Communication, Control and Computing, pp. 285-294.

18. Stefano, Marsili (1981), "Optimal Design of PID Regulators", Int. J. Control, vol. 33, No. 4, pp. 601-616.

19. Ziegler J.G., Nichols, N.B. and Rochester, N.Y. (1942) "Optimum settings for Automatic Controllers", Trans. A.S.M.E., vol. 64, pp. 759-768.

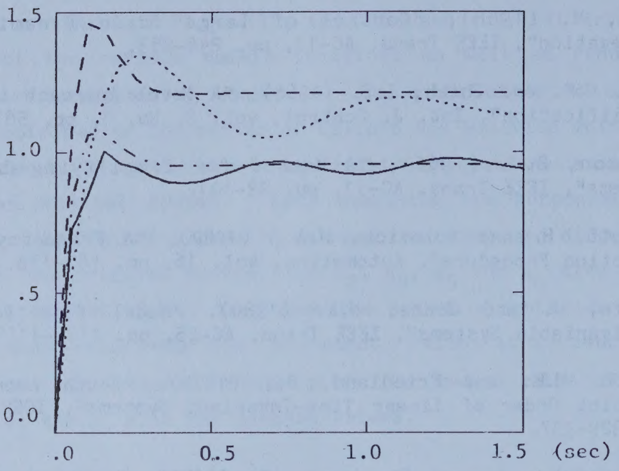


Fig. (1) Transient response of $\frac{\Delta V_t}{\Delta V_{ref}}$ due to step input.

——— original 11th order model
--------- 3rd order (Aggregation) model
-·-·-·-·- 3rd order (Aggregation with moment matching) model
.... 3rd order (singular perturbation)

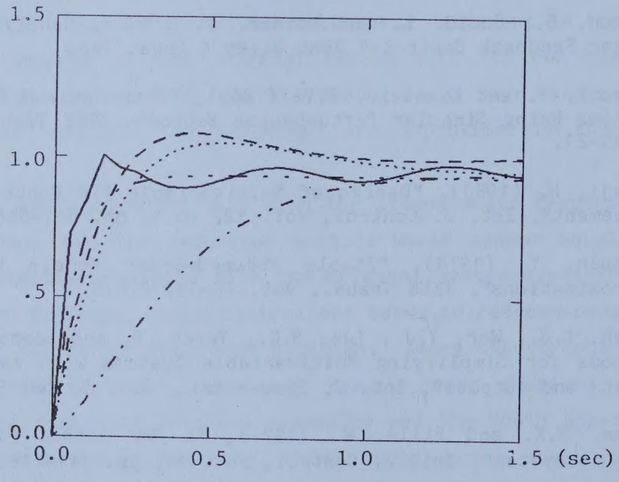


Fig. (2) Transient response of $\frac{\Delta V_t}{\Delta V_v}$ due to step input

——— original 11th order model
--------- 3rd order (Routh approx.) model
.... 3rd order (continued fraction) model
-·-·-·-·- 3rd order (Routh approx. with moment matching)
—·—·—3rd order (Elliott and Wolovich) model

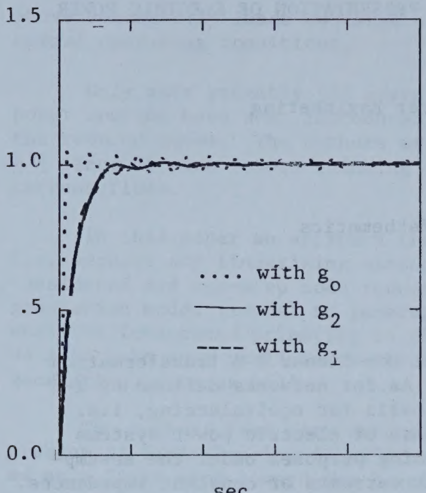


Fig. (3)

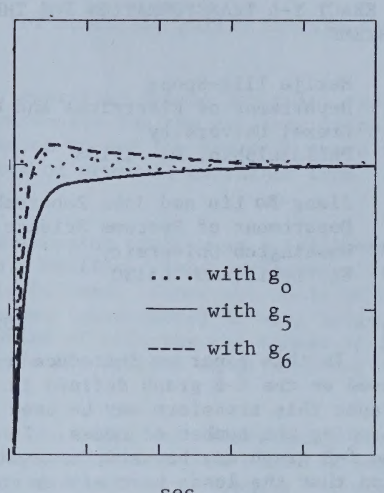


Fig. (4)

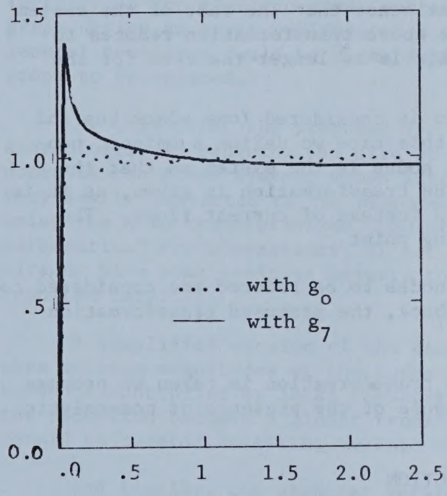


Fig. (5)

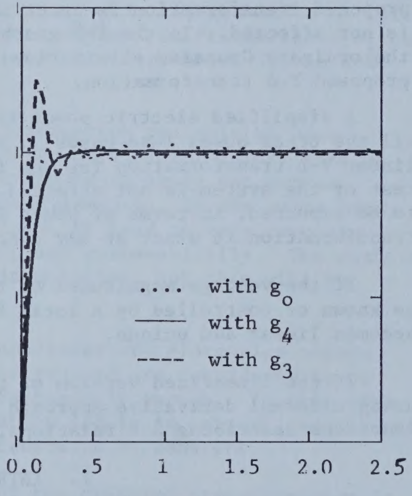


Fig. (6)

Transient response of the original 11th order system with various controllers due to step input.

AN EXACT Y-Δ TRANSFORMATION FOR THE Ŝ-Ê PRESENTATION OF ELECTRIC POWER SYSTEMS

Marija Ilic-Spong
Department of Electrical and Computer Engineering
Drexel University
Philadelphia, PA 19104

Jiang-Bo Liu and John Zaborszky
Department of Systems Science and Mathematics
Washington University
St. Louis, MO 63130

ABSTRACT

In this paper we introduce a direct non-linear Y-Δ transformation based on the Ŝ-Ê graph defined in [7]. As for networks defined by Î-Ê graphs this transform may be used as a basis for equivalencing, i.e. reducing the number of nodes. In the case of electric power systems the Î-Ê graph may be used for equivalencing purposes under the assumption that the loads have either constant currents or constant impedences. Using the Ŝ-Ê graph this assumption is no longer necessary. Any functional dependence of the power flows through the primitive elements on voltages applied to their terminals may be modeled exactly. The proposed transformation is exact in the sense that the rest of the system is not affected. In the Î-Ê graph the above transformation reduces to the ordinary Gaussian elimination. This is no longer the case for the proposed Y-Δ transformation.

A simplified electric power system is considered (one slack bus and all the other buses load buses). For this case we define a unique, non-linear Y-Δ transformation for any four nodes in the system so that the rest of the system is not affected. The transformation is given, as it is to be expected, in terms of power flows instead of current flows. The transformation is exact at any operating point.

If the voltage magnitudes of the nodes to be reduced are considered to be known or controlled by a local feedback, the proposed transformation becomes linear and unique.

If the linearized version of this transformation is taken we propose using a formal derivative approach because of the presence of nonanalytic functions describing Ŝ-Ê relations.

I. INTRODUCTION

In the large electric power system, which may have close to 2000 nodes and almost twice as many transmission lines, a common practice is to use so called equivalencing techniques [2,9] to reduce the system order. The standard approach to this problem is often to assume constant values for all nonlinear elements in the system and to treat the entire problem as a linear system relating currents and voltages. In this way the system reduction is effected through a classical Y-Δ (or star-mesh) transformation [1]. However, as it is strongly emphasized in [2] most of the loads (and generators as well) are highly nonlinear and hence the linear approximation

---

This research was supported by the Department of Energy under contract numbers DEAC01-78-ET29636, DE-AC01-79-ET29367 and by NSF under grant INT-7902976.

above necessarily leads to large errors for operating points outside the normal operating conditions.

Only very recently [5] state (i.e. operating point) dependent reactive power sources have been introduced to compensate for the nonlinearities of the reduced nodes. The authors were pleased to see the approach taken in [3] since it also favors thinking in terms of power flows rather than current flows.

In this paper we approach the equivalencing problem as a nonlinear one, i.e. without any linearizing assumptions. Nonlinear load performance is considered and one-step node reduction is defined. Since the ratio of non-generation nodes (loads) to generation nodes (generators) is very large, we shall be interested primarily in the problem of reducing the number of loads. We also take some preliminary steps towards the problem of eliminating generating units as well.

The authors feel very strongly that the right way of attacking the problem of equivalencing is as a nonlinear problem and this paper is intended as a first step toward the development of a recursive formula for step by step node reduction.

In this paper we give a very brief overview of the power flow-voltage ($\hat{S}$-$\hat{E}$) graph for electric power systems introduced in [4] and based on this graph derive an exact nonlinear Y-Δ transformation. Different cases of the general technique could be given depending on a set of nodes in the $\hat{S}$-$\hat{E}$ graph to be reduced.

At this point the proposed Y-Δ transformation can only be used sporadically in reducing the size of a system. All the Y (star) sets of nodes in the given system, whose intersection is an empty set can be very easily subjected to the proposed transformation. On the other hand, using the same transformation sequentially might lead to very complex mathematical transformations, if not performed systematically. The authors already have some positive answers to this question, but they will be reported later.

A simplified version of the exact nonlinear transformation occurs when voltage magnitudes at the nodes to be reduced are considered to be known or controlled by local feedback. Fortunately enough, in this case the reduction becomes a linear transformation and it is unique. Again, one should be careful in taking voltage magnitudes to be constant.

And finally, the simplest version of the proposed transformation is introduced after linearizing the flow equations around any operating point, using formal partial derivatives. This is a well known mathematical concept [5] but it has only recently been introduced for linearization purposes into electric power systems [6]. The benefits of this approach are still yet to come from the practical point of view. Because it allows one to treat load flow equations on a complex domain C^n, the size of the problem is cut in half in comparison to the solution done on R^{2n}. It should be emphasized that this linearization does not lead to the same equations as a classical Y-Δ transformation. Both the mathematical transformation and the state variables used are quite different than the one given in [1].

II. NOTATION

$j \overset{\Delta}{=} +\sqrt{-1}$

R^n - n dimensional real space

C^n - n dimensional complex space

$\hat{E}_i = E_{ix} + jE_{iy}$	rectangular presentation of the voltage at the i^{th} node
$\hat{E}_i = E_i \angle \hat{E}_i$	polar coordinates presentation of the voltage at the i^{th} node
$\hat{E}_i^*$	complex conjugate of $\hat{E}_i$
E_i	voltage magnitude of $\hat{E}_i$
$\angle \hat{E}_i$	phase angle of $\hat{E}_i$
$\hat{Y}_{ik}$	admittance of the transmission line connecting nodes i and k
$\hat{Y}_{ik} = jB_{ik}$	admittance of the lossless transmission line connecting nodes i and k
$\hat{S}_{ik}^T = \hat{E}_k(\hat{E}_i - \hat{E}_k)^* \hat{Y}_{ik}^*$	completely transmitted power from node i to node k
$\hat{S}_{ik}^L = \lvert\hat{E}_i - \hat{E}_k\rvert^2 \hat{Y}_{ik}^*$	loss power for the line connecting nodes i and k
$\hat{S} = P + jQ$	given flow through the load
$\frac{\partial \hat{f}}{\partial \underline{\hat{x}}} \triangleq \left(\frac{\partial \hat{f}}{\partial \underline{x}_x} - j\frac{\partial \hat{f}}{\partial \underline{x}_y}\right)/2$	formal partial derivative of a complex function $\hat{f}$ with respect to the vector $\underline{\hat{x}}$ of complex variables $\hat{x}$
$\frac{\partial \hat{f}}{\partial \underline{\hat{x}}^*} \triangleq \left(\frac{\partial \hat{f}}{\partial \underline{x}_x} + j\frac{\partial \hat{f}}{\partial \underline{x}_y}\right)/2$	formal partial derivative of a complex function $\hat{f}$ with respect to the complex conjugate vector $\underline{\hat{x}}^*$.
$\underline{\hat{x}} = \underline{x}_x + j\underline{x}_y$	column vector of complex variables $\hat{x}$
o.p.	operating point

III. AN EXACT Y-Δ (STAR-MESH TRANSFORMATION)

In order to present this, a very short review of the $\hat{S}$-$\hat{E}$ graph presentation of an electric power system [4] is given. For more details see [7,8].

The most common primitive elements present in an electric network are (current-voltage $\hat{I}$-$\hat{E}$ graph presentation):

1. (i) Ideal voltage sources
   (ii) Ideal current sources
   (iii) R, L and/or C elements connecting (i) and (ii).

The most common primitive elements present in an electric power system are ($\hat{S}$-$\hat{E}$ graph presentation):

2. (i) Ideal voltage sources (slack bus(es))
   (ii) Ideal power sources (loads)
   (iii) Transmission elements (lines, AC-DC converters, tap changing transformers).

One of the reasons for working in terms of power flows is the fact that for an ideal current (voltage) source one cannot prespecify the voltage (current) on its ends. Therefore, ideal power sources 2.(ii) do not have a proper presentation in the $\hat{I}$-$\hat{E}$ graph. The $\hat{S}$-$\hat{E}$ graph, as it has been explained in [4,7] requires a different model for the primitive elements of power systems: Kirchhoff's preservation law for the power flows through them has to be satisfied, instead of for current flows. This way the basic network theory laws applied to the $\hat{S}$-$\hat{E}$ graph can be written in terms of $\hat{S}$ and $\hat{E}$ vector spaces, instead of the conventional $\hat{I}$ and $\hat{E}$ vector spaces. For these purposes transmission lines are modeled as L sections in the $\hat{S}$-$\hat{E}$ graph as shown in Figure 1.

A completely transmitted power from node i to node k is given as

$$\hat{S}_{ik}^{T} = \hat{E}_k(\hat{E}_i - \hat{E}_k)^* \hat{Y}_{ik}^* \tag{1}$$

The loss power from node i to node k is

$$\hat{S}_{ik}^{L} = |\hat{E}_i - \hat{E}_k|^2 \hat{Y}_{ik}^* \tag{2}$$

To each node in an electric power system there corresponds one branch in the $\hat{S}$-$\hat{E}$ graph connected between the ground and the node where the load power is injected. Since the power flow through these branches is prespecified (both its real and reactive part) they represent ideal power flow sources. This allows any functional dependence of flow with respect to the voltage difference across these branches:

$$\hat{S}_i = \hat{f}_i(\hat{E}_i) \tag{3}$$

To the slack bus there corresponds one branch in the $\hat{S}$-$\hat{E}$ graph representing an ideal voltage source.

Some of the possible Y (star) combinations in the $\hat{S}$-$\hat{E}$ graph, whose central node is a subject to reduction are (Figure 3):

(i) Two branches presenting completely transmitted power, one branch presenting losses.

(ii) Two branches presenting completely transmitted powers, two branches presenting losses.

(iii) Two transmitted power branches, one loss branch and one ideal power flow source.

It is possible to detect many others. The above mentioned are just the typical one. The proposed techniques, slightly modified, apply to all of them.

The process by which nodes are eliminated is similar to the classical star-mesh transformation [1]. The basic principle is that the injected power flows into the retained nodes and voltages applied to them are the same as before the reduction. This principle will provide the necessary equations for our transformation. The idea is to write Kirchhoff's flow law for the node which we want to eliminate and calculate from this relation that voltage in terms of the other three voltages. For example, on Figure 2

$$\hat{E}_3 = \hat{f}_3\ (\hat{E}_1, \hat{E}_2, \hat{E}_o) \tag{4}$$

Then from the requirement that the injected power flows into nodes 1 and 2 are the same as before the reduction, get a new power flow through the Δ configuration. It so happens that there is one more degree of freedom than necessary. Therefore, we propose two solutions:

(1) Assume that $\hat{S}_{10}{}^{t} = \hat{S}_{20}{}^{t}$ (symmetry)

(2) Instead of Δ structure get an asymmetric L structure on either node 1 or 2.

In this paper we formulate only case (1). We also actually derive equations for the third case (iii), namely two transmitted powers, one loss power and one ideal power source (Figure 2c). For symmetry reasons we assume

$$\hat{S}_{10}{}^{t} = \hat{S}_{20}{}^{t} \tag{5}$$

The voltage $\hat{E}_3$ is determined from Kirchhoff's flow equation for the node to be eliminated:

$$-\hat{S}_{1,3}{}^{T} + \hat{S}_{3,2}{}^{T} + \hat{S}_{3,2}{}^{L} + \hat{S} = 0 \tag{6}$$

which together with the expressions for the transmitted and loss power (1), (2) gives

$$-\hat{E}_3(\hat{E}_1-\hat{E}_3)^{*}\hat{Y}_{1,3}{}^{*} + \hat{E}_2(\hat{E}_3-\hat{E}_2)^{*}\hat{Y}_{2,3}{}^{*} + |\hat{E}_3-\hat{E}_2|^2\hat{Y}_{2,3}{}^{*} + \hat{S} = 0 \tag{7}$$

Moreover, for lossless transmission lines

$$\hat{Y}_{ik} = jB_{ik} \tag{8}$$

and (7) becomes

$$E_{3x}\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right) - E_{3y}\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right) + P = 0 \tag{8a}$$

$$E_{3y}\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right) + E_{3x}\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right) - (E_{3x}{}^{2}+E_{3y}{}^{2})\left(\sum_{i=1}^{2} B_{i3}\right) + Q = 0 \tag{8b}$$

Now the technique is to solve first from a linear equation (8a) for E_{3y} in terms of E_{3x} and then solve the quadratic equation (8b) for E_{3x}. In a conventional per unit system E_{3x} is always close to 1 p.u. and non-negative. This condition naturally determines the choice for the sign in (8b). Then E_{3y} is just a linear combination of E_{3x} and there is no difficulty in choosing its sign:

$$E_{3y} = \frac{E_{3x}\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right) + P}{\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)} \tag{9}$$

and

$$E_{3x} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{10}$$

with

$$a = -\left(\sum_{i=1}^{2} B_{i3}\right)\left(\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)^2 + \left(\sum_{i=1}^{2} E_{iy}B_{i3}\right)^2\right) \tag{11}$$

$$b = \left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right)^2 + \left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)^3 - 2P\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right)\left(\sum_{i=1}^{2} B_{i3}\right)) \tag{12}$$

$$c = Q\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)^2 - P\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right) + P\left(\sum_{i=1}^{2} E_{ix}B_{i3}\right)\left(\sum_{i=1}^{2} E_{iy}B_{i3}\right) \tag{13}$$

Once the voltage

$$\hat{E}_3 = \hat{f}_3(\hat{E}_1,\hat{E}_2,\hat{E}_o) \tag{14}$$

is determined from (8a), (8b), the power flow $\hat{S}_{12}{}^{t}$, $\hat{S}_{10}{}^{t} = \hat{S}_{20}{}^{t}$ is determined by satisfying

Kirchhoff's flow law at node 1:

$$\hat{S}_{12}{}^{t} + \hat{S}_{10}{}^{t} = \hat{S}_{13}{}^{T} \tag{15}$$

and Kirchhoff's flow law at mode 2:

$$-\hat{S}_{12}{}^{t} + \hat{S}_{10}{}^{t} = -\hat{S}_{32}{}^{T} \tag{16}$$

which gives

$$\hat{S}_{10}{}^{t} = \tfrac{1}{2}(\hat{S}_{13}{}^{T} - \hat{S}_{32}{}^{T}) = \hat{f}_{10}(\hat{E}_1,\hat{E}_2,\hat{f}_3(\hat{E}_1,\hat{E}_2,\hat{E}_o),\hat{E}_o) \tag{17}$$

and

$$\hat{S}_{12}{}^{t} = \tfrac{1}{2}(\hat{S}_{13}{}^{T} + \hat{S}_{32}{}^{T}) = \hat{f}_{12}(\hat{E}_1,\hat{E}_2,\hat{f}_3(\hat{E}_1,\hat{E}_2,\hat{E}_o),\hat{E}_o) \tag{18}$$

Now (17) and (18) completely characterize the new Δ structure in terms of the remaining voltages, so that node 3 is eliminated.

All the other possible Y combinations in the $\hat{S}$-$\hat{E}$ graph would follow the above pattern.

The most interesting case is, still, reducing a generator node; the magnitude of this node is known. Kirchhoff's flow balance for the real power defines the phase angle of the voltage at this node uniquely up to its sign. To the authors knowledge this is the first exact formula for eliminating generator nodes.

(7) in this case has a form

$$\mathrm{Re}(-\hat{E}_3(\hat{E}_1-\hat{E}_3)*\hat{Y}_{13}{}^{*} + \hat{E}_3(\hat{E}_3-\hat{E}_2)*\hat{Y}_{23}{}^{*}) + P = 0 \tag{19}$$

and with $E_3 = 1$ p.u. (20)

$$\hat{E}_3 = (\cos\theta_3 + j\ \sin\theta_3) \tag{21}$$

It is known that for the stable electric power systems

$$|\theta_3| \le \frac{\pi}{2} \tag{22}$$

and therefore

$$\cos\theta_3 = +\sqrt{1 - \sin^2\theta_3} \tag{23}$$

which gives from (19)

$$(\sum_{i=1}^{2} E_{ix}B_{i3})\ \sqrt{1-\sin^2\theta_3} - (\sum_{i=1}^{2} E_{iy}B_{i3})\ \sin\theta_3 + \frac{P}{E_3} = 0 \tag{24}$$

This gives a straightforward solution for $\sin\theta_3$ up to the sign.

IV. Y-Δ TRANSFORMATION FOR CONSTANT VOLTAGE MAGNITUDES

A simplified version of this nonlinear transformation is given when voltage magnitudes at the nodes to be reduced are known or are controlled by a local feedback, which is very often the case.

Fortunately enough, in this case the proposed technique becomes a linear transformation. It is also always unique.

Again, for the standard per unit system in electric power systems we take

$$E_i^2 = 1 \text{ p.u.} \tag{25}$$

Then for lossless transmission lines equation (7) becomes

$$\begin{aligned} a_{11}E_{3x} + a_{12}E_{3y} &= b_1 \\ a_{21}E_{3x} + a_{22}E_{3y} &= b_2 \end{aligned} \tag{26}$$

where

$$a_{11} = (\sum_{i=1}^{2} E_{iy}B_{i3}) \tag{27}$$

$$a_{12} = -(\sum_{i=1}^{2} E_{ix}B_{i3}) \tag{28}$$

$$a_{21} = -a_{12};\ a_{22} = a_{11} \tag{29}$$

$$b_1 = -P;\ b_2 = (\sum_{i=1}^{2} B_{13}) - Q \tag{30}$$

It is easy to solve (26) as a system of two linear equations in two unknowns. The solution for (26) is always unique since

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \neq 0 \tag{31}$$

is always satisfied.

V. THE LINEARIZED VERSION OF THE Y-Δ TRANSFORMATION

Using a notion of formal partial derivatives [7,8] for the linearization of the complex function written for the Kirchhoff flow law at the node to be reduced (Figure 3c)

$$\hat{f} = -\hat{S}_{13}^{T} + \hat{S}_{32}^{T} + \hat{S}_{32}^{L} + \hat{S} = 0 \tag{32}$$

or

$$\hat{f} = \hat{f}(\hat{E}_1, \hat{E}_2, \hat{E}_3) \tag{33}$$

gives

$$\frac{\partial \hat{f}}{\partial \underline{\hat{E}}}\bigg|_{o.p.} = \left(\begin{array}{l} \dfrac{\partial \hat{f}}{\partial \hat{E}_1} = 0 \\[2ex] \dfrac{\partial \hat{f}}{\partial \hat{E}_2} = 0 \\[2ex] \dfrac{\partial \hat{f}}{\partial \hat{E}_3} -\hat{E}_1^*\hat{Y}_{13}^* - \hat{E}_2^*\hat{Y}_{23}^* + E_3^*(\hat{Y}_{13}^* + \hat{Y}_{23}^*) \end{array}\right)\bigg|_{o.p.} \tag{34}$$

and

$$\frac{\partial \hat{f}}{\partial \underline{\hat{E}}^*}\,_{o.p.} = \left(\begin{array}{lcl} \dfrac{\partial \hat{f}}{\partial \hat{E}_1^*} & = & -\hat{E}_3\hat{Y}_{1,3}^* \\[2ex] \dfrac{\partial \hat{f}}{\partial \hat{E}_2^*} & = & -\hat{E}_3\hat{Y}_{23}^* \\[2ex] \dfrac{\partial \hat{f}}{\partial \hat{E}_3^*} & = & \hat{E}_3(\hat{Y}_{1,3}^* + \hat{Y}_{23}^*) \end{array}\right)\bigg|_{o.p.} \tag{35}$$

(32), (34) and (35) together give

$$\hat{f}\big|_{o.p.} + d\hat{f}\big|_{o.p.} = \hat{S} + \left(\frac{\partial \hat{f}}{\partial \underline{\hat{E}}}\right)^T\bigg|_{o.p.} (\hat{E}-\hat{E}\big|_{o.p.}) + \left(\frac{\partial \hat{f}}{\partial \underline{\hat{E}}^*}\right)^T\bigg|_{o.p.} (\hat{E}^*-\hat{E}^*\big|_{o.p.})$$

$$= 0 \tag{36}$$

In particular the linearization around the flat start ($E_i = 1$ p.u., $\angle\hat{E}_i = 0$, $\forall i$) gives

$$\hat{S} + \begin{pmatrix} 0 \\ 0 \\ (\hat{Y}_{13}^* + \hat{Y}_{23}^*) \end{pmatrix} \begin{pmatrix} (\hat{E}_1-1) \\ (\hat{E}_2-1) \\ (\hat{E}_3-1) \end{pmatrix} + \begin{pmatrix} -\hat{Y}_{13}^* \\ -\hat{Y}_{23}^* \\ (\hat{Y}_{13}^* + \hat{Y}_{23}^*) \end{pmatrix} \begin{pmatrix} (\hat{E}_1^*-1) \\ (\hat{E}_2^*-1) \\ (\hat{E}_3^*-1) \end{pmatrix} = 0 \tag{37}$$

and (37) leads to the elimination of node 3:

$$\hat{E}_3^* = \frac{\hat{S} + \hat{Y}_{13}^*\hat{E}_1 + \hat{Y}_{23}^*\hat{E}_2}{(\hat{Y}_{1,3}^* + \hat{Y}_{23}^*)} \tag{38}$$

The corresponding power flows through the Δ configuration follow from (17) and (18).

VI. CONCLUSIONS

In this paper we introduce based on the novel Ŝ-Ê graph for electric power systems [6] a direct nonlinear Y-Δ transformation. As it is known for Î-Ê graphs, this transformation is a basis for equivalencing, i.e. reducing the number of nodes. This graph has also been used for equivalencing purposes in electric power systems under the assumption that the loads have either constant currents or constant impedances. Using the $\hat{S}$-$\hat{E}$ graph this assumption is no longer necessary. Any functional dependence of the power flows through the primitive elements on voltages applied to their terminals may be modeled exactly.

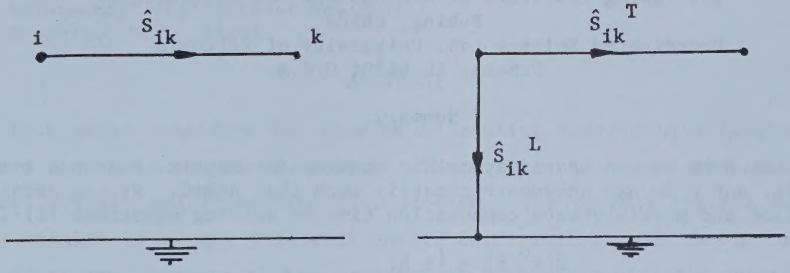
The proposed Y-Δ transformation is exact in the sense that the rest of the system is not affected. For given voltage magnitudes the proposed transformation simplifies to the linear one.

If the linearized version of this transformation is taken we propose using a formal derivative approach because of the presence of nonanalytic functions describing $\hat{S}$-$\hat{E}$ relations.

The authors feel very strongly that the right way of attacking the problem of equivalencing is as a nonlinear problem and this paper is intended as a first step toward the development of a recursive formula for step by step node reduction.

VII. REFERENCES

1. Hostetter, G.H., Fundamentals of Network Analysis, Harper & Row, N.Y., 1980.
2. Gelopulos, D., Lightfoot, S., "Realistic Reduction of Nonlinear Networks," IEEE Trans. Power Apparatus and Systems, vol. PAS-97, 1978, pp. 495-510.
3. Van Ameroogen, R.A.M., Van Meeteren, H.P., "A Generalized Ward Equivalent for Security Analysis", IEEE PES Summer Meeting, Portland, Oregon, July 1981, #81SM 455-5.
4. Ilic-Spong, M., "A New Approach to Load Flow Studies of Electric Power Systems Using Complex Power-Complex Voltage Graphs", D.Sc. Thesis, Washington Univ., St. Louis, MO, December 1980.
5. Fuks, B.A., Introduction to the Theory of Analytic Functions of Several Complex Variables, Providence, Rhode Island: American Mathematical Society, 1963.
6. Bandler, J.W., El-Kady, M.A., "A Novel Formulation for Steady-State Sensitivity Analysis in Power Networks", Proc. of the IEEE ICCC, Hague, The Netherlands, 1980.
7. Ilic-Spong, M., Zaborszky, J., "A Different Approach to the Load Flow Problem", Proc. of 20th PICA Conference, Philadelphia, 1981, pp. 18-26.
8. Zaborszky, J., Ilic-Spong, M., Whang, K.W., "A Homogeneous, Nonlinear Load Flow Algorithm for HV-AC and Compound HV-AC-DC Systems", to be presented at IEEE PES Winter Meeting, N.Y., 1982.
9. Snyder, W.L., "Load Flow Equivalent Circuits - An Overview", IEEE PES Winter Meeting, New York, 1972.

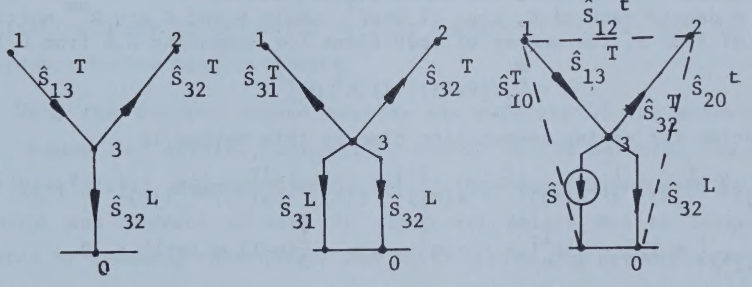


i
$\hat{S}_{ik}$
k
$\hat{S}_{ik}^{T}$
$\hat{S}_{ik}^{L}$

a) A transmission line connecting nodes i and k

b) The L section for the transmission line a)

Figure 1



1
2
3
0
$\hat{S}_{13}^{T}$
$\hat{S}_{32}^{T}$
$\hat{S}_{32}^{L}$
$\hat{S}_{31}^{T}$
$\hat{S}_{31}^{L}$
$\hat{S}_{12}^{t}$
$\hat{S}_{13}^{T}$
$\hat{S}_{10}^{T}$
$\hat{S}_{20}^{t}$
$\hat{S}_{32}^{T}$
$\hat{S}$
$\hat{S}_{32}^{L}$

(i)

(ii)

(iii)

Figure 2

AN EFFICIENT METHOD OF SOLUTION FOR NEARLY SYMMETRIC MATRICES

Y. F. Zhou
The Peking Institute of Aeronautics and Astronautics
Peking, China
Coordinated Science Lab. University of Illinois
Urbana, IL 61801 U.S.A.

Summary

Let A be an nxn nearly symmetric nonsingular matrix, B an nxn symmetric matrix, and C an nxn unsymmetric matrix such that A=B+C. We can obtain the solution and possibly save computation time by solving equations (1)-(3) [2].

$$B|x^{o}\ X| = |b\ H| \tag{1}$$

$$(I_m + K^{T}X)\omega = K^{T}x^{o} \tag{2}$$

$$x = x^{o} - X\,\omega \tag{3}$$

The following properties and results have been obtained.

Properties of matrix C:

1. Any pair of entries c_{ij} and c_{ji} of C are both zero or at least one of them is zero.
2. Let $a_{ij}-a_{ji}=k$, then A can be expressed as a symmetric matrix plus a nonsymmetric matrix C where either $c_{ij}=k$ and $c_{ji}=0$, or $c_{ij}=0$ and $c_{ji}=-k$. The choice of C affects the computational time.
3. Let m denote rank of C, then if $C=HK^{T}$, where H and K are R^{nm} matrices and of rank m, the number of operations for computing H.K from C is

$$\omega = \sum_{j=1}^{m} (|H(*j)|-1)\,|K^{T}(j*)|^{*}$$

The criterion for saving computation time by this method is

$$\sum_{j=1}^{n}[|L_{(*j)}||U^{I}_{(j*)}| - \tfrac{1}{2}|L^{I}_{s(*j)}|^{2} - |L_{s(*j)}||Y_{(j*)}| - |L^{IT}_{s(*j)}||X_{(j*)}| - |K^{T}(*j)|$$
$$(|X_{(j*)}| + 1)] - \tfrac{3}{2}|L^{I}_{s}| - \omega - m^{2} - n.m - \tfrac{1}{3}(m-1)\,m\,(m+1) \geq 0 \tag{4}$$

Where the subscript s is for the symmetric matrix B, Y is the solution of of $L_sY=H$ and X the solution of $L^{IT}_{s}X=Y$. If A is structurally symmetric, and m is a small number, and Y.X are full matrices, then the criterion becomes

$$\tfrac{1}{2}\sum_{j=1}^{n}|L^{I}_{s(*j)}|^{2} - (2m+0.5)|L^{I}_{s}| - n\,m\,(m+2) \geq 0$$

References

[1] I. N. Hajj, "Sparsity Consideration in Network Solution by Tearing", IEEE Circuit Syst., Vol CAS-27, pp. 357-366, No. 5, May 1980.

[2] G. Guardabassi and A. Sangiovanni-Vincentelli, "A Two Level Algorithm for Tearing," IEEE Trans. Circuit Syst., Vol CAS-23, pp. 783-791, No. 12, Dec. 1976.

---

[*] All symbols are the same as in [1].

DECENTRALIZED DYNAMIC CLEARING OF CONGESTED MULTI-DESTINATION NETWORKS*

P. E. SARACHIK
Department of Electrical Engineering
Polytechnic Institute of New York
Brooklyn, N.Y. 11201

ABSTRACT

This paper considers the problem of routing traffic with many destinations through a congested network. The problem is formulated as a very general dynamic multicommodity flow problem. The optimal control problem which results is a very difficult one to solve analytically (even for open loop solutions) because of the presence of both state variable constraints and delays in the system dynamics.

The difficulties inherent in the centralized problem can be avoided by using a decentralized approach to obtain dynamic routing strategies. Some of the results obtained using this approach are presented.

INTRODUCTION

Consider a network made up of N nodes connected by L directed links along which the rate of traffic flow can be controlled. M of the nodes are destination nodes toward which varying amounts of traffic must be routed in accordance to the traffic demand.

When the traffic demand exceeds the capacity of the network to meet this demand for service, congestion builds in the network. In the model used here (first suggested by Gazis[1]) the congested network is viewed as a store and forward network in which all delays due to congestion are treated by assuming that queues can build inside the network at the network nodes. Traffic is assumed to move between nodes at a constant velocity (this corresponds a fixed travel time between nodes) and is stored at the nodes in queues until it can proceed onward toward its destination.

The degree of congestion is measured by the aggregate of all queues in the network. The problem considered here is that which occurs after a "rush hour" or period of heavy demand when the demand has fallen below the capacity but congestion is still present.

Using the store and forward model the queue dynamics can be expressed

---

*The research was supported by the National Science Foundation under grant ENG-77-14898, and partially by NATO grant #1425.

as

$$\dot{q}_n^m(t)=r_n^m(t) - \sum_{k=1}^{N} u_{nk}^m(t) + \sum_{j=1}^{N} u_{jn}^m(t-s_{jn}^m) \qquad (1)$$

for $n=1,\ldots,N$
$m=1,\ldots,M$
and $n \neq m$.

where q_n^m denotes a continuous state variable which approximates the length of the traffic queue at node n having node m as its destination; u_{nk}^m denotes the rate of traffic flow to destination m on link (nk); r_n^m is the traffic demand entering at n with destination m and s_{jn}^m is the delay on link (j,n) encountered by traffic to m. For deterministic input demand, the objective is to clear the congestion (so that $q_n^m(T)=0$ for all n and m) at a specified time T which is large enough to achieve this objective while minimizing the aggregate cost of congestion

$$J\{\underset{\sim}{u}\}= \int_{t_0}^{T} \sum_{m=1}^{M} \sum_{n=1}^{N} [q_n^m(t) + \sum_{k=1}^{N} s_{nk}^m u_{nk}^m(t)]dt \qquad (2)$$

In this integral the q_n^m terms give the aggregate delay due directly to the congestion while the flow terms produce a cost or aggregate delay associated with the link flows. The constraints to be met

$$q_n^m(t) \geq 0; \; u_{nk}^m(t) \geq 0 \text{ and } \sum_{m=1}^{M} u_{nk}^m(t) \leq C_{nk} \qquad (3)$$

for all m, n, k and t

impose the requirements that queue lengths and flows must be non-negative and that each link has a finite capacity C_{nk}.

D'Ans and Gazis[2] have shown, that an optimal open loop control for this centralized control problem can be computed by discretizing the time appropriately. When this is done, the problem can be expressed as a linear programming problem and an optimal open loop solution can be computed for any initial conditions.

This problem with all s_{nk}^m equal to zero was considered by Moss and Segall[3]. They present a set of necessary conditions via the maximum principle but they offer a general procedure for finding the flows in feedback form, only for single destination networks with zero input demand.

In seeking a closed loop solution to this centralized optimal control problem, one runs into considerable difficulty because even necessary conditions are not available for problems having both state variable constants and time-delay systems.

A DECENTRALIZED APPROACH

Most of the difficuties inherent in the centralized formulation can be avoided by adopting a decentralized approach to this problem. In general the decentralized approach to controlling large scale systems is to use a number of low level controllers operating on local information to control portions of a large system. This local information is supplemented by some interchange of information between local controllers or between local controllers and a supervisory controller. How well the overall system operates, depends on what information is exchanged. The local controllers are designed to perform their limited tasks optimally. For the dynamic routing problem considered here, the local controllers are node level controllers which must assign traffic flow on the links leaving the node based on the information available locally.

The Local Optimization Problem

The optimization problem faced at a typical node is based on the queue dynamics

$$\dot{q}^m(t) = R^m(t) - \sum_{\ell=1}^{\bar{L}} u_\ell^m(t) \tag{4}$$

where R^m is the total rate of traffic arrivals (from both upstream nodes and from outside the network) which have the destination m and there are $\bar{L}$ links which exit from the node. We wish to find the link flows $u_\ell^m(t)$ as a function of the local state vector $\underline{q}(t)$, to minimize

$$J\{\underset{\sim}{u}\} = \int_{t_0}^{T} \sum_{m=1}^{M} [q^m(t) + \sum_{\ell=1}^{\bar{L}} S_\ell^m u_\ell^m(t)]dt \tag{5}$$

where S_ℓ^m denotes the cost or delay per unit of flow on link ℓ to reach destination m. The constraints are as in (3) and $\underline{q}\ (T) = \underline{0}$. It is apparent that since all incoming demand has been consolidated into one term, the local controller is not attempting to control a time-delay system. A feedback solution can be thus be sought using Hamilton-Jacobi theory. The minimum cost due to congestion can be expressed as

$$V(\underline{q}(t0)) = \min_{\underset{\sim}{u}} \{J\{\underset{\sim}{u}\}-J^*\} = \min_{\underset{\sim}{u}} \int_{t_0}^{T} \sum_{m=1}^{M} [q^m(t) + \sum_{\ell=1}^{\bar{L}} S_\ell^m (u_\ell^m(t)-u_\ell^{m*}(t))dt \tag{6}$$

where u_ℓ^{m*} are the optimal non-congested flows (i.e. the flows which minimize J in (5) when $\underline{q}\ (t_0) = \underline{0}$) and J* is the corresponding minimum non-congested cost on $[t_0, T]$. Note that $R^m(t) = \sum_{\ell=1}^{\bar{L}} u_\ell^{m*}(t)$. When the

R^m are constants, then the minimum cost function $V(\underline{q})$ satisfies the Hamilton-Jacobi equation

$$\min_{\underline{u}} \sum_{m=1}^{M} [q^m(t) + \sum_{\ell=1}^{\bar{L}} (S_\ell{}^m - \frac{\partial V}{\partial q^m})(u_\ell{}^m(t) - u_\ell{}^{m*})] = 0 \tag{7}$$

where $V(\underline{0}) = 0$ and the minimization is subject to constraints as in (3). If the $V(\underline{q})$ which satisfies (7) and boundary conditions imposed by state constraints can be found then this would produce necessary and sufficient conditions for the optimal flows.

A complete solution to the local optimum control problem is not yet available. A method for obtaining an approximate solution was presented by Özgüner and Sarachik[4]. However the exact solution is known only for two special cases.

Two Special Cases

The Single Destination Case (M=1). Chu[5] first solved this local dynamic routing problem using a geometric approach. Later Sarachik and Özgüner[6] obtained the same solution using the approach above. Since (7) must be minimized at each time instant, this is equivalent to choosing the $u_\ell(t)$ to minimize

$$\sum_{\ell=1}^{\bar{L}} (S_\ell - \frac{\partial V}{\partial q})\, u_\ell(t) \tag{8}$$

subject to the constraint $0 \leq u_\ell(t) \leq C_\ell$. This says that we should choose

$$u_\ell(t) = \begin{cases} C_\ell & \text{for } S_\ell < \dfrac{\partial V}{\partial q} \\ 0 & \text{for } S_\ell \geq \dfrac{\partial V}{\partial q} \end{cases} \tag{9}$$

Thus if the links are numbered according to the ordering $S_1 \leq S_2 \leq \ldots \leq S_L$ and if K is the largest index ℓ satisfying $S_\ell < (\partial V/\partial q)$ we see that on links $\ell=1,2,\ldots,$ K the full capacity should be used to carry traffic flow whereas on links $\ell=K+1,\ldots,$ L no traffic should flow. Substituting this result in (8) gives

$$u_\ell(t)_{opt} = \begin{cases} C_\ell & \text{for } q(t) > Y_\ell \\ 0 & \text{for } q(t) \leq Y_\ell \end{cases} \tag{10}$$

where the thresholds Y_ℓ are given by

$$Y_\ell = \sum_{i=1}^{\ell} (C_i - u_i^*)(S_\ell - S_i) \tag{11}$$

for $\ell=1,\ldots\bar{L}$.

Multi-destination with L=2. Sarachik[7] obtained a solution to (7) with arbitrary M, for a node with only two exit links. The method of solution involved finding the solutions along the state space axes (these are one dimensional problems) and using the results as boundary conditions for the admissible state space $\underline{q} \geq \underline{0}$. The derivation can be found in [7]. The results show that the admissible state space is divided into non-intersecting regions by planes. In each region the optimal flows must satisfy a necessary and sufficient condition. The regions can be found by simple threshold tests on aggregate queue lengths. Specifically with $\beta_m \triangleq S_2{}^m - S_1{}^m$ (the additional cost of using link 2) the destinations are ordered according to $\beta_1 \geq \beta_2 \geq \ldots \geq \beta_K \geq 0 \geq \beta_{K+1} \geq \ldots \geq \beta_M$ (note that traffic to destinations 1 and K prefer to use link 1). Thresholds are calculated using

$$X_1(k) = X_1(k+1) + \gamma_1(k)\,[\beta_k - \beta_{k+1}] \text{ for } k = K, \ldots, 1 \tag{12}$$

with $X_1(K+1) \triangleq \beta_{K+1}\,\gamma_1(K)$ and

$$X_2(k) = X_2(k-1) + \gamma_2(k-1)\,[\beta_{k-1} - \beta_k] \text{ for } k = K+1, \ldots, M \tag{13}$$

with $X_2(K) \triangleq -\beta_K \gamma_2(K)$. The $\gamma_i(k)$ denotes the capacity available on link i for emptying queues when the demand R^m for destinations m=1 to k is routed on link 1 and the rest on link 2 (note that for k ≠ K this means some traffic will by diverted away from its preferred link). The composite queue lengths are next defined as

$$Q_1(k) \triangleq \sum_{m=1}^{k} q^m \quad ; \quad Q_2(k) \triangleq \sum_{m=k+1}^{M} q^m \tag{14}$$

and the test variable $T_1(k) = (\gamma_2(k)Q_1(k) - \gamma_1(k)Q_2(k))/\gamma_2(k)$ is found for all those $k \leq K$ which also make $\gamma_2(k) > 0$). For $k \leq K$ if $X_1(k+1) < T_1(k) \leq X_1(k)$ and $Q_1(k)$, $Q_2(k) > 0$ the optimum flows satisfy

$$\begin{aligned} &\text{a)} \quad U_1(k) = C_1;\ u_1{}^m = 0 \text{ for } m > k \\ &\text{b)} \quad U_2(k) = C_2;\ u_2{}^m = 0 \text{ for } m \leq k \end{aligned} \tag{15}$$

where the composite flows are defined as

$$U_1(k) \triangleq \sum_{m=1}^{k} u_1{}^m \quad ; \quad U_2(k) \triangleq \sum_{m=k+1}^{M} u_2{}^m \tag{16}$$

whereas if $T_1(k-1) \leq X_1(k) < T_1(k)$ the optimum flows must satisfy (15a) and

$$U_2(k) + u_2{}^k = C_2 \quad ; \quad u_2{}^m = 0 \text{ for } m \leq k-1. \tag{17}$$

The only difference between (15b) and (17) is that (17) permits a non-zero u_2^k. For a state on the boundary of the admissible region (i.e. when $Q_i(k)=0$ for some k or i) the conditions (15) and (17) must be modified slightly. In this case the $U_i(k)$ must equal the corresponding aggregate demand.

If the above tests fail then for k > K the test variable $T_2(k) = (\gamma_1(k)\ Q_2(k) - \gamma_2(k)Q_1(k))/\gamma_1(k)$ is formed for all those k for which $\gamma_1(k) > 0$. For $Q_1(k)$, $Q_2(k) > 0$ if $X_2(k) < T_2(k) \leq X_2(k+1)$ the optimum flows must satisfy (15) whereas if $T_2(k) \leq X_2(k) < T_2(k-1)$ the optimum flows satisfy (15a) and (17). It should be noted that (15) or (17) do not specify the flows uniquely so many optimal implementations are possible.

Coordination of Local Control

In order for the node level controllers to implement the local control strategies, information must be supplied to it about the parameters S_ℓ^m. This can be done for each link exiting the node by forming $S_\ell^m = s_\ell^m + \phi_j^m$ where s_ℓ^m is the delay (or cost) of traversing link ℓ to the first downstream node j and ϕ_j^m denotes an estimate of the delay (or cost) of reaching destination m from node j. These ϕ_j^m cost-to-go terms must be computed at each node and transmitted to the upstream nearest neighbor. The way these terms are calculated greatly affects the performance of the overall network. Chu[5] using a capacity weighted average for a single destination network (M=1) defined

$$\phi_j^m(t) = \frac{q_j^m(t) + \sum_{\ell=1}^{\bar{L}} S_\ell^m C_\ell}{\sum_{\ell=1}^{\bar{L}} C_\ell} \tag{18}$$

and had to restrict the application to non-cyclic networks. Whereas if C_ℓ is replaced by the excess capacity $\gamma_\ell \triangleq C_\ell - u_\ell^*$ the strategy can be extended to networks containing cyclic paths since it can be proven that no cyclic flows will occur[6]. Sarachik[8] using flow weighted averages (i.e. replace C_ℓ in (18) by u_ℓ^m) for the multi-destination case required the non-cyclic restriction. This restriction was removed in [4] by using a certain kind of excess capacity average. The question of what is the best coordination scheme is still open.

CONCLUSION

This paper has described the general problem of using dynamic routing to clear congestion from a congested network. A decentralized strategy was described which utilizes local node level controllers. The optimal local strategies were presented for 2 special cases when traffic demand is constant. Research is being done to extend the local results for traffic demand modeled by a Poisson arrival process and to obtain the exact optimum routing for M destinations with L exit links. More research is also needed to determine whether an optimal coordination strategy can be found.

REFERENCES

[1] Gazis, D.C., "Modelling and optimal control of congested transportation systems", Networks, Vol. 4, pp. 113-124, (1974).

[2] G.C. D'ans and D.C. Gazis, "Optimal Control of Oversaturated Feed - Forward Transportation Networks," IBM Research Report, #RC4543; 1973.

[3] F.H. Moss and A. Segall, "An Optimal Control Approach to Dynamic Routing in Data-Communication Networks," Report EE312. Faculty of EE, Technion; 1977 (To appear in IEEE Trans. Aut. Control).

[4] Özgüner, Ü. and Sarachik, P.E., "Decentralized Routing in Congested Multi-Destination Traffic Networks" Proc. of 1981 ICCC Conference, pp. 1006-1009; 1981.

[5] Chu, K.C., "Decentralized real-time control of congested traffic networks", IBM Research Report, RC-6337, Dec. 1976.

[6] Sarachik, P.E. and Özgüner, Ü., "On decentralized routing in traffic networks," Univ. of Toronto, Dept. of Electrical Engineering, Systems Control Report #8003, April 1980. To appear in IEEE Trans. Aut. Cont.

[7] Sarachik, P.E., "Optimal local dynamic routing for clearing multi-destination neworks," EE Dept. Report. Polytechnic Inst. of N.Y.; Jan. 1981.

[8] Sarachik, P.E., "Clearing of congested multi-destination networks," Proc. of Engr. Found. Conf. on Research Dev. in Comp. Control of Urban Traffic Syst., Feb. 1979.

ON THE DECENTRALIZED CONTROL OF AN AUTOMATED HIGHWAY

ROBERT E. MAGEE ROBERT E. FENTON
Department of Electrical Engineering
The Ohio State University
2015 Neil Avenue
Columbus, Ohio 43210

ABSTRACT

An automated highway system (AHS) is one approach toward alleviating future traffic problems. AHS control would be accomplished by a computer hierarchy and should be characterized by,

i) Control flexibility (especially under emergency conditions),
ii) An efficient distribution of the computational load,
iii) Minimal communications,
iv) Minimal trip-time delay and trip-time variance, and
v) Energy efficiency.

One widely studied structure involves a synchronous approach to vehicle management. This approach will be described and an efficient means for controlling a complex network will be delineated.

INTRODUCTION

A proposed partial solution to many of the problems associated with congested, intercity highways is a network consisting of automated and nonautomated lanes. Dual-mode vehicles (i.e., those capable of both manual and automated operation) would be employed resulting in a simpler evolution from today's highways and greater flexibility for a user.

A typical scenario might be as follows:

> A vehicle would be driven manually to an automated system entry point where it would undergo a rapid checkout. If it failed this checkout, the vehicle would be directed to a nearby service facility; otherwise, it would move to an entrance ramp. Here the driver would enter the desired destination into an onboard computer, the vehicle would be automatically guided up the ramp and into the traffic stream, and proceed under automatic control until the selected exit were reached. Then, the driver would regain control and manually proceed to the final destination.

The potential benefits of such an Automated Highway System (AHS), include an increase in safety, more efficient use of present highway right-of-ways, modest fuel economies, and a reduction in driver stress.

NETWORK CONTROL POLICIES

Three approaches to network control have been suggested for high-speed, small time-headway, AHS operations -- an asynchronous policy, a synchronous policy, and a quasi-synchronous one. The asynchronous, or car-following, approach involves vehicular control with respect to the state of the surrounding traffic. The earliest paper dealing with such control was presented by Zworykin and Flory (1) in 1958; subsequently, numerous others have been presented in the literature; e.g. (2)-(4). This approach, as commonly envisioned, utilizes vehicles with the capability of making complex control decisions. One unsolved problem is the resolution of vehicle conflicts at merging junctions where the coordinated control of many vehicles would be essential. Here, the vehicles would arrive at a junction and then try to resolve any conflicts. This could lead to junction saturation and trip-time delay (i.e., any excess traffic may have to stop). Even if these disadvantages were acceptable, the required vehicle-based technology for achieving such control is unavailable.

The synchronous, or point-follower, approach is characterized by deterministic vehicular control. This involves the concept of slots (of H_t in time units) moving through a network so that slot synchronization is achieved at all merging junctions. This approach was first suggested by Gluck (5) and has subsequently been studied in detail by many others including Godfrey (6), Boyd and Lucas (7), and Howson (8).

A synchronous approach, where a vehicle's entire trajectory (i.e., its schedule over a selected route) is computed prior to its entry into the system, could be called "predeterministic". Here a sequence of slot assignments, which would allow an uninterrupted trip from origin to destination, would be specified for each vehicle. Control would be centralized in a large computer and relatively "dumb" vehicles would be employed. A high-speed computer, programmed with the appropriate algorithms, would calculate optimum trajectories (based on fuel economy, trip-time variance, percent of loading, or whatever performance criterion were desired) for every vehicle. While this approach might be the best from a system point of view (i.e., the system is completely observable and completely controllable), the practical problems of implementation would be very difficult.

These problems include:

i) Deficiencies in system utilization (i.e., vehicles might be denied entry because a conflict-free trajectory could not be determined when, in fact, a trajectory could be created with the proper maneuvering of online vehicles).

ii) An inability to respond effectively to vehicle and system failures (i.e., a failure could necessitate the rescheduling and/or rerouting of many vehicles).

A less rigid and more adaptive approach is to remove the requirement for a priori slot assignments. Instead, a vehicle would be merged into mainline traffic, possibly into the first available empty slot, and then moved in synchronization with other traffic until a conflict point (i.e., a merging junction) were reached; there maneuvering might be required to effect safe and efficient operation. This "postdeterministic" approach was introduced by Munson (9), and subsequently extended by numerous investigators including Rule (10), Roseler et al (11), Kornhauser and McEvaddy (12), Whitney (13), and Sarachik and Chu (14). This policy involves minimal waiting time at an entry point, a possible distribution of the computational load, and an enhanced ability to handle system anomalies.

The quasi-synchronous approach is a combination of the asynchronous and synchronous policies. Vehicles would operate autonomously (i.e., under vehicle-follower control) on the open highway, and dependently (i.e., via point-follower control by a wayside computer) near a high-speed merging junction. This approach is similar to the postdeterministic one except for the added complexity of resynchronizing the vehicles near merges (maintaining synchronization throughout the network would not be as difficult).

The evidence for the superiority of one policy over another is not conclusive, and the published results appear to be dependent upon the choice of network configuration, its geographical extent, and the nature of the imposed travel demands. It does appear however, that greater operating flexibility, less travel delays, and a simpler implementation result from a postdeterministic synchronous policy; therefore, this choice is made here.

SYSTEM CONFIGURATION OVERVIEW

A postdeterministic synchronous approach could be realized using a 4-level, hierarchical control structure consisting of network, regional, sector, and vehicle controllers (see Fig. 1).

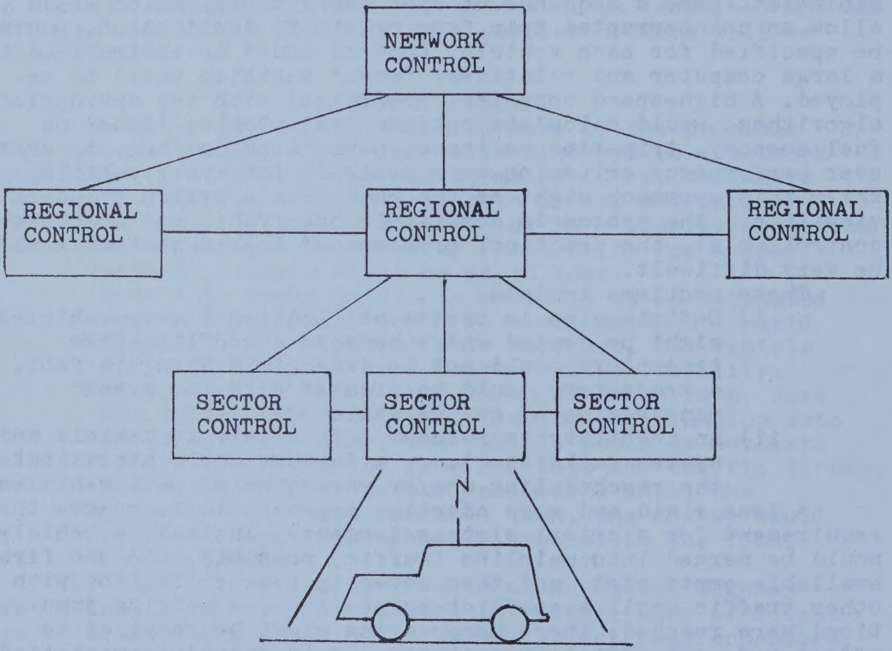


Fig. 1

The top level, a network controller which would know the desired state of every vehicle, would maintain frequently updated statistics on the network state. Such information, in conjunction with historical data from past operations, would be employed to control vehicle entry, routing, and scheduling both under normal and certain abnormal conditions. This would be a macroscopic level of control with the derived commands being transmitted to the appropriate regional level. Basically, this controller would be responsible for coordinating the activities of the various regional controllers.

Each regional controller would function as a "mini" network controller. Command decisions would be based on "gross" commands from the network computer, data from adjacent regional computers, and data from the sector computers within its jurisdiction. The regional controller's duties would include: distributing network commands, modified to the extent required by local conditions, to the appropriate sectors; providing an information source for the network computer; and managing certain types of failures.

Each sector controller, based on information received from its regional controller and the vehicles within its jurisdiction, would be responsible for maintaining slot synchronization, making slot assignments to avoid conflicts at traffic junctions, specifying vehicle trajectories, and determining abnormal conditions.

Each vehicle, upon receiving its trajectory, would act so as to maintain the desired lateral and longitudinal states. Its controller would optimize longitudinal control to achieve passenger comfort and fuel efficiency within the bounds of safety and the availability of unoccupied adjacent slots. When the adjacent slot(s) were occupied, the vehicle would maintain tight control of its position and velocity (15); if unoccupied, looser control would be employed (note however, that even though the vehicle could be allowed the latitude to "move" around, it would "know" its assigned point and could move there when ordered by the sector controller.

NOTATION AND TERMINOLOGY

The postdeterministic approach described here is based on the concept of cycles which has been extended to include variable length cycles. A cycle is a group of adjacent slots which are considered as a single entity when a decision concerning flow-rates or merging must be made.

A convenient way to represent the vehicles and their locations for purposes of slot assignments is by means of matrices of 0's and 1's, where a 1 represents a vehicle and a 0 an empty slot. The matrix is a representation of a cycle where a row indicates the status of a particular lane and a column indicates the status of a "coincident set". A typical merge with the corresponding cycle is shown in Fig. 2.

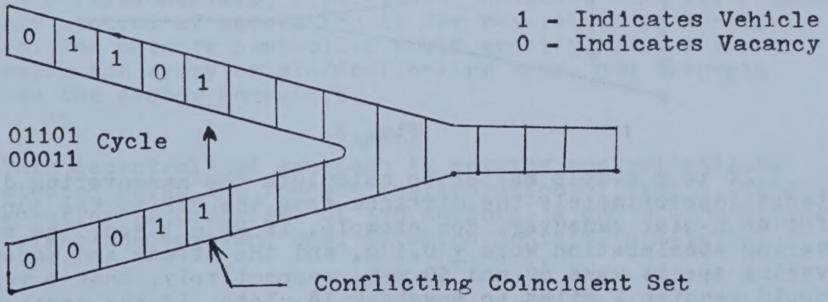


Fig. 2

The formation of the cycles is a crucial part of this approach. When a conflict is found (i.e., more than one vehicle in a coincident set), a cycle would be started. Each succeeding coincident set would be checked and a running tally of the number of vehicles versus the number of sets would be kept. When the number of vehicles equaled the number of sets, or the number of sets reached an allowed maximum (M_m), the cycle would be ended; e.g., the cycle of Fig. 2, where $5 \leq M_m$, contains 5 coincident sets.

THE MERGING PROBLEM

The most difficult normal (nonemergency) control problem, which must be continuously solved in an AHS, is the safe and efficient merging of two or more lanes of high-speed traffic. This problem has two principal aspects -- vehicle conflicts and junction saturation. The former occurs when two or more vehicles occupy upstream slots that will merge into one downstream slot, and the latter when the vehicle flow-rate into a junction exceeds the slot flow-rate out.

The resolution of conflicts would be accomplished by mainline maneuvering (i.e., moving vehicles forward or backwards a finite number of slots). Here, movebacks only are employed in the interest of fuel economy and enhanced safety. These maneuvers would be specified by the sector computer, which would examine the strings of vehicles entering a merging sector and calculate the appropriate commands for transmission to the vehicles.

A typical implementation for a sector containing a 2-into-1 merging junction is shown in Fig. 3. As the vehicles enter the sector and pass an Information Acquisition Point (IAP), the sector computer would note which vehicles were in conflict, calculate the appropriate commands, and transmit them to the vehicles at the Command Transmission Point (CTP). The vehicles would then begin the required maneuvering, which would be completed before the upcoming junction were reached.

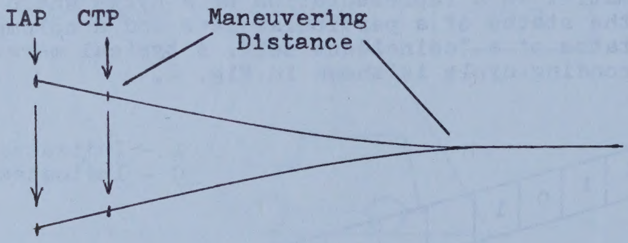


Fig. 3

It is a simple matter to calculate the maneuvering distance (approximately the distance from the CTP to the junction) for an n-slot maneuver. For example, if $H_t = 1$ sec, the maneuvering acceleration were ± 0.1 g, and the stream and maneuvering speeds were 60 and 50 mph, respectively, then a vehicle would require 2 miles to moveback 14 slots. If the assignment algorithm for conflict resolution were such that the minimum number of vehicles were to be moved, the worse-case maneuver would consist of moving a vehicle from the beginning of a cycle to the end (i.e., a 13-slot moveback).

If a cycle had not ended when the maximum length (14 slots in this example) were reached, the junction would be saturated. This could arise either from too many vehicles in too few slots, or from the approach taken to conflict resolution. The latter results primarily from the limited amount of maneuvering distance available.

The cycle shown in Fig. 4 is 15 slots long (i.e., 15 vehicles in 15 coincident sets) and thus could not be resolved in a sector where $M_m = 14$. Even though a vacancy exists in slot 15, the vehicle in slot 1 of lane B could not moveback the necessary 14 slots in the maneuvering space available; i.e., the junction would, in effect, be "saturated".

0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Lane A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Lane B
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

Fig. 4

Some proposed solutions to this problem are:

i) The slowing down, and possible stopping of some conflicting vehicles until a vacant slot occurs;[1]

ii) Diverting the "excess" vehicles into waiting areas; and

iii) Rerouting some vehicles onto the adjacent nonautomated highways.

With the possible exception of the last, these approaches would probably lead to excessive trip-time delays and trip-time variance.

Another approach (the one adopted here) would be for the regional controller to identify the cycles for an upcoming merge and order whatever vehicle maneuvers would be necessary prior to the cycle entering the sector. The same limitation would apply at this level as at the sector level i.e., a finite maneuvering distance would be available. If more vehicles were to enter a region than could be maneuvered to avoid sector saturation, regional saturation would result. The latter must be prevented by the network controller.

In a fixed-headway, fixed-route, constant line speed AHS, the only control of saturation is the regulation of vehicle entries. The network controller would specify allowed input flow rates for every origin/destination pair, and transmit these to the sector computers.

EXAMPLE

This decentralized approach to merging control will be demonstrated using the roadway configuration of Fig. 5. This configuration is comprised of two sectors, **A** and B, consisting of straight roadways and one, **C**, which contains a 2-into-1 merging junction. The first step in the merging process for this junction would be performed by the network controller; i.e., it would ensure that the regional computer would be able to create cycles of no more than M_m slots.[2] The regional controller would create cycles from the lanes in Sectors A and B and verify that all cycles were less than M_m slots.

---

[1] Any maneuvering speed less than stream speed could be selected; however, as this quantity is decreased, there is an increasing likelihood, especially in high-density situations, of disturbances propagating upstream.

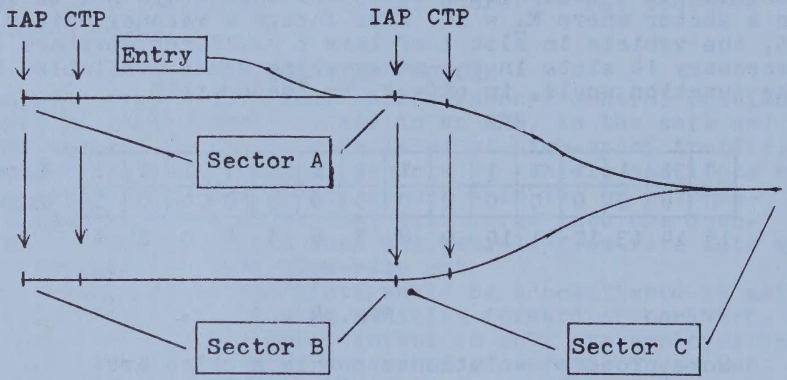


Fig. 5

An example of cycle formation is shown in Fig. 6 where $M_m = 6$. The 4th cycle would cause saturation unless the regional computer ordered Sector A or B, (or both) to maneuver, in this case, two vehicles. An efficient algorithm for this purpose is readily derived.

In some instances, Sector C could be extended to encompass a larger maneuvering distance and therefore an ability to service longer cycles. However, if one or more upstream merging junctions were sufficiently close to this sector, a geographical limit on such an expansion would exist. In addition, additional entry point(s) would be acquired for this sector to control, and the length of controlled roadway would increase. At some point, the added complexity, which is directly related to the local geometrics, would pose a limit on sector size.

Lane A 0 0 1 1 1 1 1 1 1 0 1

Lane B 0 0 0 0 0 0 1 1 0 0 0

1st cycle 1 0

2nd cycle 0 0

3rd cycle 1 0

4th cycle 0 0 1 1 1 1 1 1 / 0 0 0 0 0 0 1 1 (8 vehicles in 8 coincident sets)

Fig. 6

---

[2] M_m could be a different number at each junction as it is dependent on the available maneuvering distance.

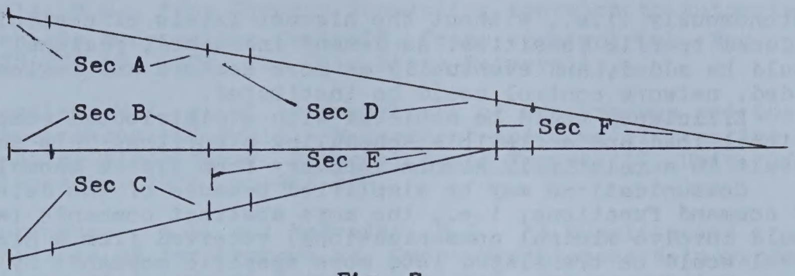


Fig. 7

A more complex region is shown in Fig. 7. Here sectors E and F each contain a merging junction. The regional controller would examine sectors B and C and issue the appropriate commands so that saturation could be avoided in Sector E. However, this controller would also have to examine Sectors D and E in conjunction with A,B, and C, to provide appropriate vehicle commands so that saturation could be avoided in Sector F.

If, as is shown in Fig. 8, two merging junctions were so close as to prevent the formation of a reasonable-sized sector, the junction would have to be combined into one sector comprised of (at least for control purposes) a 3-into-1 merge.

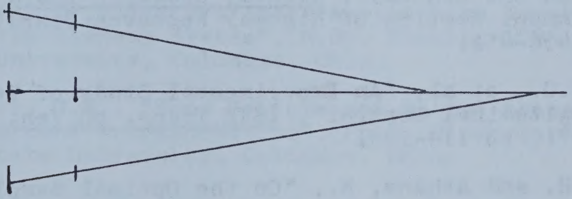


Fig. 8

In the configuration of both Figs. 7 and 8, the geometrics pose a severe constraint on the control problem, and illustrate one reason why the control cannot be fully decentralized. Such geometrics are similar to those that would be encountered in the most critical area, the intracity highway, and therefore must be considered if realistic results are to be obtained.

SUMMARY

An AHS should employ a control hierarchy that is characterized by:

i) Control flexibility;
ii) Utilization efficiency;
iii) Minimal communications;
iv) Minimal trip-time delay and trip-time variance; and,
v) Energy efficiency.

The approach presented is consistent with these criteria.

This approach is flexible as it could be applied to the merging geometrics commonly encountered in a highway environment. In addition, it could be deployed in simple stages. For example, several sectors could be initially operated autonomously (i.e., without the highest levels of control) at reduced traffic densities. As demand increased, regional control could be added, and eventually as more sectors and regions were added, network control could be instituted.

Efficiency would be achieved with a distributed computational load and a flexible scheduling algorithm. This should result in a relatively simple recovery from system anomalies.

Communications may be simplified because of the delegation of command functions; i.e., the more abstract commands (which would involve minimal communications) received from a higher level would be translated into more specific commands by a lower-level controller.

By avoiding internal queues and the unnecessary exiting of vehicles, trip-time variance should be a minimal value. Also, the network control functions should maintain system utilization at a high efficiency, and thus minimize average trip-time delay.

Some vehicle fuel efficiency would be achieved by maintaining an efficient choice of stream speeds, and the avoidance of unnecessary maneuvers.

Much effort remains in this area, and hopefully some insights for further work have been provided in this paper.

REFERENCES

1. Zworykin, V.K. and Flory,"L.E., Electronic Control of Motor Vehicles on the Highway", Highway Research Board (Proceedings of 37th Annual Meeting of Highway Research Board), Vol. 37, 1958, pp 436-451.

2. Bender, J.G., et al, "An Experimental Study of Vehicle Automatic Longitudinal Control", IEEE Trans. on Veh. Tech., VT-20, 1971, pp 114-123.

3. Levis, A.H. and Athans, M., "On the Optimal Sampled-Data Control of a String of Vehicles", Transportation Science, Vol. 2, 1968, pp 362-382.

4. Bender, J.G. and Fenton, R.E., "On the Flow Capacity of Automated Highways", Transportation Science, Vol. 4, 1970 pp 52-63.

5. Gluck, H.D., "Some Design Criteria for an Automated Roadway", M.Sc. Thesis, 1964, M.I.T. Cambridge, Mass.

6. Godfrey, M.B., "Merging in Automated Transportation Systems", Sc.D Thesis, 1964, M.I.T. Cambridge, Mass.

7. Boyd, R.K. and Lucas, M.P., "How to Run an Automated Transportation System", IEEE Transactions on Systems, Man, and Cybernetics, SMC-2, 1972, pp 331-341.

8. Howson, L.L., "Computer Simulation for an Automated Roadway Network", Gen. Motors Res. Lab. Rep. GMR-1153, 1972.

9. Munson, A.V., "Quasi-Synchronous Control of High-Capacity PRT Networks", Personal Rapid Transit, Univ. of Minnesota, 1972, pp 325-350.

10. Rule, R.G., "The Dynamic Scheduling Approach to Automated Vehicle Macroscopic Control", Trans. Control Lab. Rep. EES-276-18, 1974, The Ohio State University.

11. Roseler, W.J. et al, "Comparisons of Synchronous and Quasi-Synchronous PRT Vehicle Management and Some Alternative Routing Algorithms", Personal Rapid Transit II, Univ. of Minnesota, 1974, pp 425-438.

12. Kornhauser, A.L. and McEvaddy, P.J., "A Quantitative Analysis of Synchronous vs. Quasi-Synchronous Network Operations of Automated Transit Systems", Transportation Research, Vol. 9, 1975, pp 241-248.

13. Whitney, D.E., "A Finite State Approach to Vehicle Merging", Journal of Dynamic Systems, Measurement, and Control, June 1972, pp 147-151.

14. Sarachik, P.E. and Chu, K.C., "Real-Time Merging of High-Speed Vehicular Strings", Transportation Science, Vol. 9, Number 2, May, 1975 pp 122-138.

15. Fenton, R.E., "A Headway Safety Policy for Automated Highway Operations", IEEE Transactions on Vehicular Technology, Vol. VT-28, No. 1, Feb 1979.

16. Rakocy, L.C., "Systems*level Approach to Merging in an Automated Highway System", M.Sc. Thesis, 1980, Ohio State University, Columbus, Ohio.

17. Takasaki, G.M., "Some Vehicle and Sector Aspects of Automated Ground Transportation", Ph.D. Dissertation, 1977, Ohio State University, Columbus, Ohio.

COMMUNICATION CONTROL FOR LOCAL NETWORKS

ROBERT S. PRINTIS

Xerox Corporation
Office Products Division
Palo Alto, California

INTRODUCTION

Computer communication networks have begun to appear commercially in numerous configurations, providing services for various applications in distributed computing and data processing. One of the most attractive forms for these networks has been as a loosely coupled set of local networks, consisting of large long-haul networks interconnecting the set of local networks. This architecture has particular appeal to office automation applications where office buildings, or campuses, may be individually "wired" with an independent local computer communications network, but interconnected into a larger network using available public communications facilities. An example of a successful implementation of this technology has been the internetwork of Ethernets implemented in the Xerox development network [Boggs80] and the Xerox 8000 Network System. Local networks are distinguised by their area coverage (usually a single building or campus of buildings), their channel access method, and their speeds.

The Ethernet [Metcalfe76] is an example of a local network. We shall pay particular attention to the communication control problem presented by this class of local network. In this paper, we discuss systems theory aspects which are useful in understanding the operation of these loosely coupled systems.

REVIEW OF ETHERNET PRINCIPLES

The approach used in Ethernet can be traced to the radio-based Aloha packet switching network, developed at the University of Hawaii [Abramson70]. In this network, terminals are equipped with packet radios and share a common multiaccess radio channel. A significant point is that this system has no central controller responsible for the allocation of channel resources to the terminals. Instead a *random access* procedure was devised which allowed the terminals, in a decentralized manner, to determine when they can transmit on the channel. In this scheme, two terminals which decide to transmit simultaneously produce a *collision*. After a collision, each terminal waits for a random period of time and then retransmits. This procedure continues until the packet is sucessfully transmitted.

It is important to note that the Aloha network terminals' control decisions are not based on any knowlegde of the current state of the system. This fact led to many studies on the stability and load-carrying capacity of the network [Kleinrock75]. The Ethernet is among a class of packet networks which is similar in function to the Aloha network, but uses a different control mechanism and information structure. When an Ethernet Station wishes to transmit a packet, it first determines if the channel is currently being used by sensing "carrier" on the channel. If there is a station currently transmitting, it *defers* its transmission until it hears that the channel is idle. If no other station is transmitting, it immediately begins transmission. Thus it's information structure is constructed so that the current state of the channel influences the control decision.

This channel state information is used by Ethernet stations in a control strategy which is local to the station and is based both on the channel information and a global system policy. Note that the goal of the station is to minimize its delay, or repsonse time, where delay is defined as the elapsed time from when the packet is added to the transmit queue until it is successfully received by the receiving station. Due to this decentralized control policy, it is possible for two or more stations to simultaneously transmit packets onto the channel, thus causing a *collision.* This is due to the signal propagation delay between the two stations. Another station with a packet in the transmit queue which is "down-stream" from the transmitting station may start to transmit before the "carrier" arrives at its receiver.

The control law is as follows:

> (C1) If a packet is in the transmit queue and the channel is sensed to be idle, transmit the packet, otherwise defer transmission until the channel is idle, then transmit the packet,
>
> (C2) while transmitting, continue to monitor the channel. If the signal heard is not the same as that which is being transmitted, abort the transmission and transmit a *collision consensus enforcment* signal (jam). Wait for a random time interval and repeat C1.

The jam signal is used to ensure that all transmitting stations have sensed the collision. Like the Aloha system, transmitting stations waits for a random time interval before it retransmits its packet. The algorithm determining the random interval, called the *backoff algorithm,* is such that the retransmission interval is a function of the channel load. The maximum amount of the backoff, B_{max}, is defined by the rule:

$$B_{max} = 2^n$$

where n denotes the number of times that the station has attempted to retransmit a collided packet.

With this rule the colliding station's policy is to randomly select a time interval between 0 and B_{max}, then wait for that interval of time to elapse before attempting to retransmit the packet. The channel loading information, then, is embedded in the value n above. It indexes by one for each retransmission attempt, therefore providing a measure of the number of stations with packets ready to be sent on the network. The effect of the policy is to spread the demand for the channel over time, resulting in a higher average individual station delay when the channel is heavily loaded, but improving global system performance measures such as fairness in allocating the channel capacity, while maintaining channel throughput.

This system has been implemented on a coaxial cable media and been under investigation for a number of years, at both commercial and academic institutions. A system of 3Mbit Ethernets have been in use at Xerox Corporation since 1975. At this time, there are approximately 50 Ethernets of this design with approximately 1500 computers, connected via the public telephone network, world-wide. The performance of the 3Mbit Ethernet has been reported elsewhere [Shoch 1980]. A specification for an Ethernet with 10 Megabit per second capacity was published by Digital Equipment Corporation, Intel Corporation and Xerox Corporation in September, 1980 [Ethernet80]. The purpose of this short paper is to place the results of this experience in the context of systems theory.

STRUCTURE OF THE SYSTEMS PROBLEM

The local network system consists of individual stations attached to a common coaxial cable. Each Ethernet station is viewed as an independent decision maker with access to certain information concerning the shared communication channel. This information space is the triple {idle, busy, contention}. The channel, then, may be in one of three states, idle (no station desires to transmit), busy (exactly one station has acquired the channel), or contention (different stations are simultaneously attempting to transmit).

The state space of the decision maker's system is the couple {idle, blocked}. A station is "blocked" if it has collided during a previous transmission attempt and is waiting to retransmit the packet. In this model a user station's transmit buffer can hold precisely one packet.

Its control space is analogous to traffic light controls, {green, red}. If the station has the green light, it may immediately transmit a ready packet, red light, it cannot transmit even though it has a ready packet.

The Ethernet control policy is that all stations must implement an adaptive feedback control policy, as in B_{max} above, which performs (C1) and (C2) above. The system, stations integrated into the network, has the goal of maximizing system throughput, where throughput is given in [Metcalfe76] is given by

$$E = T_p/(T_p + (W^*T_s))$$

where T_p is the packet transmission time in seconds, T_s is the maximum roundtrip propagation delay in seconds (usually called *slot time*) and W is the waiting time distribution for a system in which it is assumed that there is constant demand for the channel. It is given by the geometric distribution

$$W = (1 - A)/A$$

$$A = (1 - (1/Q))^{(Q-1)}$$

where Q is the level of demand for the channel. It represents the number of stations ready to transmit on the Ethernet. In fact, using the above the following table is obtained for a 10 megabit per second Ethernet:

Q	Tp	Ts	Emax
256	1.2ms	51.2µs	.93
256	820µs	51.2µs	.90
256	410µs	51.2µs	.82
256	102µs	51.2µs	.54
256	51µs	51.2µs	.37

The system throughput, or E, is an instantaneous measurement of the load being carried by the network when demand is at a level Q. This translates to the fraction of the channel capacity available, when needed, while providing fair access to stations wishing to transmit. The system goal is to maximize E over various values of Q, under the constraint that the average station delay is bounded, say below some level. The table above gives values for the maximum throughput for fixed values of Q.

The individual station's goal is to minimize its own average delay, even at the expense of the other stations on the network. If the control law (C1)/(C2) is used, the control for each station is set to green. Implying that all stations with ready packets have the permission to transmit. Under these assumptions, it has been shown in [Shoute77] that this information structure, consisting of the triple defined above and the additional information, the estimated number of ready stations, provides the optimal average delay per station with average delay proportional to

$$T_n = (1 - 1/n)^{(n-1)} - 1.$$

where n is as defined above.

In summary, systems such as Ethernet, provide the following solution to this systems problem:

> If the stations are performing locally optimum, that is if stations access the channel using the rules summarized above, and
>
> The channel provides the information required by each station, such as the number of transmitting stations,
>
> Then the system behaves optimally, i.e., the throughput is maximized while average local station delay is minimized.

There are a number of open questions concerning these distributed systems which have not been discussed by this paper. It is clear that the impact of the adaptive control policy discussed above controls the point where excessive demand, high levels of Q, can cause saturation, or instability. The dynamic behavior of Aloha-like systems was studied in [Klienrock75] and others. [Shoute77] also discusses other control schemes and information structures and its impact on system behavior. However complete dynamics of these systems is still a mystery. A study currently underway at Xerox attempts to characterize dynamic behaviour for these distributed systems with a goal of understanding such behavior as transistion to instability and recovery dynamics.

REFERENCES

[Abramson70]
Abramson, N., "The Aloha System -- Another alternative for computer communications," AFIPS Conference Proceedings (FJCC 1977), Vol. 37, 1970, pp. 281-285.

[Boggs80]
Boggs, D. R., Shoch, J. F., Taft, E. A., and Metcalfe, R. M., "PUP: An Internetwork Architecture," IEEE Transactions on Communications, COM-28:4, April, 1980, pp. 612-624.

[Ethernet80]
Intel, Digital Equipment, and Xerox Corporations, The Ethernet, A Local Area Network: Data Link Layer and Physical Layer Specifications. Version 1.0, September 30, 1980.

[Kleinrock75]
Kleinrock, L., Lam, S. S., "Packet switching in a multiaccess broadcast channel: Performance Evaluation," IEEE Transactions on Communications, COM-23, no. 4, April, 1975, pp. 410-423.

[Metcalfe76]
Metcalfe, R. M. and Boggs, D. R., "Ethernet: Distributed packet switching for local computer networks," Communications of the ACM, Vol 19, July, 1976, pp. 395-404.

[Shoch80]
Shoch, J. F. and Hupp, J. A., "Measured Performance of an Ethernet Local Network," Communications of the ACM, Vol 23, December, 1980, pp. 711-721.

[Shoute77]
Shoute, F. C., Decentralized Control in Computer Communication, Ph.D. Thesis, Harvard University, April, 1977 (Technical Report No. 667).

DISTRIBUTED REAL-TIME SIMULATION OF MULTIBODY SYSTEMS*

HOOSHANG HEMAMI
YUAN-FANG ZHENG
PAUL A. GRYGIER
Department of Electrical Engineering
The Ohio State University
Columbus, Ohio 43210

ABSTRACT

With the availability of low cost microprocessors, buses and interfaces, large dimensional mechanical and biological systems may be simulated in real time by a distributed computer system - each processor handling the dynamics of one link. The combination of a particular state space and the Newton-Euler formulation of the dynamics of rigid bodies permits this natural division of dynamics among a system of microprocessors. In this paper the formulation of dynamics of rigid bodies for multiprocessor simulation is presented via some examples.

I. INTRODUCTION

Multibody systems -- a number of interconnected rigid bodies arise in many fields. Notable among these fields are manipulators [1,2], robots [3], anthropomorphic systems for transportation safety studies [4,5,6,7] and for aiding the handicapped [8,9], locomotion systems [10,11], vehicle crash models [12], and vehicle dynamics [13,14]. For modelling of natural systems multi-body systems are extensively used. Typical examples can be cited in analysis of the knee joint [15,16,17], head and neck [18], spine [19], and a variety of normal physical human activity [20,21,22,23].

In analysis and simulation of all these systems simple derivation of the equations of motion and a convenient protocol for simulation of these systems is very desirable. The combination of a particular state space and the Newton-Euler formulation of the dynamics of rigid bodies permits a natural division of the simulation task among a system of interconnected microprocessors -- each processor simulating one rigid body. This approach should be contrasted with those discussed previously [24]. More information about derivation of the equations of motion can be found in [25,26,27,28, 29,30]. More details about the present approach can be found in [29,31,32, 33].

Consider one rigid body which may be holonomically or non-holonomically constrained. Two coordinate systems are assumed -- an inertial coordinate system and a body coordinate system whose axes are among the principal axes of the body.

Let the state variables be X, $\dot{X}$, θ, and W, where θ and W are the Euler angles and the angular velocity expressed in the body coordinate system.

The Newton-Euler state space equations of this body are

$$\dot{X} = \dot{X} \tag{1}$$

$$\ddot{X} = \frac{1}{m}(F + G) \tag{2}$$

$$\dot{\theta} = B(\theta)W \tag{3}$$

$$\dot{W} = J^{-1}[f(W) + M + N] \tag{4}$$

where J is the moment of inertia matrix, W is the angular velocity vector and where F and G are respectively applied and constraint forces expressed

---

* This work was supported in part by the National Science Foundation under Grant No. ECS 7824440 and in part by the Vehicle Research and Test Center NHTSA, East Liberty, Ohio.

in the inertial coordinate system, and M and N are respectively applied torques (due to F and G), and torques of nonholonomic constraint expressed in the body coordinate system. The 3x3 matrix $B(\theta)$ and vector $f(W)$ are well defined. Let $A(\theta)$ be the transformation from the body coordinate system to the inertial coordinate system:

$$A(\theta) = \begin{bmatrix} \cos\theta_1 & +\sin\theta_1 & 0 \\ -\sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_2 & -\sin\theta_2 \\ 0 & \sin\theta_2 & \cos\theta_2 \end{bmatrix} \begin{bmatrix} \cos\theta_3 & 0 & \sin\theta_3 \\ 0 & 1 & 0 \\ -\sin\theta_3 & 0 & \cos\theta_3 \end{bmatrix}, \tag{5}$$

$$B(\theta) = \begin{vmatrix} -\sin\theta_3/\cos\theta_2 & 0 & -\cos\theta_3/\cos\theta_2 \\ \cos\theta_3 & 0 & \sin\theta_3 \\ \tan\theta_2 \sin\theta_3 & 1 & \tan\theta_2 \cos\theta_3 \end{vmatrix} \tag{6}$$

and

$$f(W) = [w_2 w_3(j_2-j_3) \quad w_3 w_1(j_3-j_1) \quad w_1 w_2(j_1-j_2)]^T \tag{7}$$

A final notation is helpful in what follows. For an arbitrary three-dimensional vector K, $K^T = [k_1, k_2, k_3]$, the skew symmetric matrix KK is defined

$$KK = \begin{bmatrix} 0 & -k_3 & k_2 \\ k_3 & 0 & -k_1 \\ -k_2 & k_1 & 0 \end{bmatrix} \tag{8}$$

In what follows one single rigid body, an inverted pendulum, is considered. The approach is extended to a three-link manipulator in subsequent section.

II. ONE RIGID BODY

Suppose a rigid body is connected to the origin of the inertial systems. The point of contact is described by vector L in the body coordinate system. Let G be the vector of holonomic forces of constraint acting on the body at the origin, described in the inertial coordinate system. The moment of these forces with respect to the center of gravity is

$$M = LL\, A^T(\theta) G \tag{9}$$

Let Q be a constant vector in the inertial coordinate system and let C be a 3x2 matrix whose columns are orthogonal to Q

$$Q^T C = 0 \tag{10}$$

Suppose the rigid body above is further nonholonomically constrained so that W is parallel with Q:

$$W = A^T(\theta) Q \dot{\phi} \tag{11}$$

where ϕ is a quasi-coordinate. Let $\Delta^T = (\Delta_1, \Delta_2)$ be a vector whose compo- Δ_1 and Δ_2 are the nonholonomic moments of constraint with these definitions. Equations (1-4) become

$$\dot{X} = \dot{X} \tag{12}$$

$$\dot{\theta} = B(\theta) W \tag{13}$$

$$\begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix} \begin{matrix} \ddot{X} \\ W \end{matrix} = \begin{bmatrix} F \\ f(W) \end{bmatrix} + \begin{bmatrix} I & 0 \\ LLA^T & A^T C \end{bmatrix} \begin{bmatrix} G \\ \Delta \end{bmatrix} \tag{14}$$

Where I is the 3x3 identity matrix. The quasicoordinate ϕ may be eliminated and the constraints combined

$$C^T AW = 0 \tag{15}$$

$$X + AL = 0 \tag{16}$$

Equations (15) and (16) may be differentiated with respect to time to obtain

$$\begin{bmatrix} I & -ALL \\ 0 & C^T A \end{bmatrix} \begin{bmatrix} \ddot{X} \\ \dot{W} \end{bmatrix} = \begin{bmatrix} -A(WW)^2 L \\ 0 \end{bmatrix} \tag{17}$$

If $\ddot{X}$ and $\dot{W}$ are eliminated from Equations (14) and (17) quantities G and Δ are obtained as functions of the state and applied external forces. The state space remains 12 dimensional and the quasi-coordinate ϕ of Equation (11) is related to the state as follows:

$$\dot{\phi} = (Q^T Q)^{-1}\ Q^T\ AW \tag{18}$$

$$\phi = \int \dot{\phi}(t)\ dt \tag{19}$$

On the other hand G and Δ may be eliminated by multiplying Equations (14) by a row vector that is orthogonal to columns of the 6x5 matrix

$$\begin{bmatrix} I & 0 \\ LLA^T & A^T C \end{bmatrix}$$

The result is a system of interdependent differential equations that describes the constrained motion.

Suppose it is desired to reduce the dimension to six by parameterization of the 12-dimensional state space. Equation (16) is twice differentiated

$$\begin{bmatrix} \ddot{X} \\ \dot{W} \end{bmatrix} = \begin{bmatrix} -A(WW)^2 L \\ 0 \end{bmatrix} + \begin{bmatrix} ALL \\ I \end{bmatrix} \dot{W} \tag{20}$$

The coefficient of W in Equation (20) is

$$\begin{bmatrix} ALL \\ I \end{bmatrix}$$

The transpose of the above matrix:

$$\begin{bmatrix} ALL \\ I \end{bmatrix}^T = \begin{bmatrix} -LLA^T & I \end{bmatrix}$$

is orthogonal to the coefficient of G in Equation (14):

$$\begin{bmatrix} -LLA^T & I \end{bmatrix} \begin{bmatrix} I \\ LLA^T \end{bmatrix} = 0$$

This allows substitution of Equation (20) in Equation (14) and multiplying both sides by the transpose to obtain

[see Eq. (21)]

$$\begin{bmatrix} -LLA^T & I \end{bmatrix} \begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} ALL \\ I \end{bmatrix} \dot{W} \quad +$$

$$\begin{bmatrix} -LLA^T & I \end{bmatrix} \begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} -A(WW)^2L \\ 0 \end{bmatrix} =$$

$$\begin{bmatrix} -LLA^T & I \end{bmatrix} \left\{ \begin{bmatrix} F \\ f(W) \end{bmatrix} + \begin{bmatrix} 0 \\ A^TC\Delta \end{bmatrix} \right\} \qquad (21)$$

Equations (21) and (13) are the six dimensional space equations in state $Z = (\theta, W)^T$.

When it is desired to compute Δ, Equation (21) is solved for $\dot{W}$ and both sides are multiplied by C^TA to derive Δ as a function of state Z and applied force F

$$\Delta = \Delta(\theta, W, F)$$

On the other hand Δ may be eliminated by multiplying both sides of Equation (21) by Q^TA to derive a system of interdependent equations in θ and W.

A final substitution for W is from Equation (11): Differentiating Equation (11).

$$\begin{aligned} \dot{W} &= A^TQ\ddot{\phi} + \dot{A}^TQ\phi \\ &= A^TQ\ddot{\phi} - (WW)W \\ &= A^TQ\ddot{\phi} - 0 \end{aligned}$$

Therefore

$$\dot{W} = A^TQ\ddot{\phi} \qquad (22)$$

and

$$\dot{\theta} = BW = BA^TQ\,\dot{\phi} \qquad (23)$$

Under certain conditions Equations (22) and (23) may be solved for θ and W in terms of ϕ. For example, if $Q = [0\ 0\ 1]^T$

$$BA^TQ = \begin{bmatrix} \tan\theta_2 \sin\theta_1 & \tan\theta_2 \cos\theta_1 & -1 \\ \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1/\cos\theta_2 & \cos\theta_1/\cos\theta_2 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

It follows that

$$\dot{\theta}_1 = -\dot{\phi},\ \dot{\theta}_2 = 0,\ \dot{\theta}_3 = 0$$

or

$$\theta_1 = -\phi,\ \theta_2 = \theta_3 = 0 \qquad (24)$$

and

[See Equation (25)]

$$\dot{W} = \begin{bmatrix} -\sin\theta_3 \cos\theta_2 \\ \sin\theta_2 \\ \cos\theta_3 \cos\theta_2 \end{bmatrix} \ddot{\varphi} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \ddot{\varphi}$$

$$\dot{W}_1 = \dot{W}_2 = 0$$

$$\dot{W}_3 = \ddot{\varphi} = -\ddot{\theta}_1 \qquad (25)$$

Substitution of Equations (24) and (25) in Equation (21) and premultiplying both sides by $Q^T A$ to eliminate Δ results in the motion of an inverted pendulum in the plane

$$(m\ell^2 + J_3)\ddot{\phi}_1 = m\ell g \sin\phi \qquad (26)$$

In many applications it is desired to have a state space consisting of θ and $\dot{\theta}$. If equation (13) is differentiated

$$W = B^{-1}\dot{\theta} \qquad (27)$$

$$\dot{W} = B^{-1}\ddot{\theta} + \left[\sum_{i=1}^{3} \frac{\partial B^{-1}}{\partial \theta_i} \dot{\theta}_i\right] \dot{\theta} \qquad (28)$$

Equation (28) may further be manipulated to give

$$\dot{W} = B^{-1}\ddot{\theta} + \begin{bmatrix} -\sin\theta_3 & \cos\theta_3 \cos\theta_2 & -\sin\theta_3 \sin\theta_2 \\ 0 & 0 & -\cos\theta_2 \\ \cos\theta_3 & \sin\theta_3 \cos\theta_2 & \sin\theta_2 \cos\theta_3 \end{bmatrix} \begin{bmatrix} \dot{\theta}_2\dot{\theta}_3 \\ \dot{\theta}_3\dot{\theta}_1 \\ \dot{\theta}_1\dot{\theta}_2 \end{bmatrix} \qquad (29)$$

Thus W and $\dot{W}$ may be transformed to functions of θ, $\dot{\theta}$, and $\ddot{\theta}$.

III. SYSTEM OF RIGID BODIES

The state space and the transformations above can easily be extended to a system of rigid bodies. This is best demonstrated by an example. For simplicity an open chain (tree like) case is discussed.

This example is considered previously in [24,34]. Lagrangian and Newtonian dynamics were used before and the forces of constraint were eliminated. Here explicit formulas for both holonomic and non-holonomic forces are given, and a projection method, embodying the principle of virtual work, is outlined.

The system consists of three links. Link 1 rotates in the horizontal plane (x_1x_2). Link 2 and 3 have one degree of freedom joints (Fig. 1). They both move in the plane of link 1 and $x_3$axis. There are three applied torques (T_1,T_2, and T_3) at the joints. The modified Euler angles describe respectively rotations about x_3, the subsequent x_1 and finally x_2.

The principal axes of every link are labelled x, y, and z initially aligned with x_1, x_2, and x_3, respectively.

Let Γ_1, Γ_2, and Γ_3 be connection forces and Λ_1, Λ_2, and Λ_3 be the nonholonomic torques respectively at joints 1, 2, and 3.

The Newtonian acceleration equations are

$$U_1[\dot{W}_1\ \dot{W}_2\ \dot{W}_3\ \ddot{X}_1\ \ddot{X}_2\ \ddot{X}_3]^T = U_2 + U_3\ [\Gamma_1\ \Gamma_2\ \Gamma_3]^T + U_4[\Lambda_1\ \Lambda_2\ \Lambda_3]^T \tag{30}$$

the matrices U_1, U_2, U_3 and U_4 are given in the appendix.

If the three holonomic constraints are twice differentiated with respect to time one obtains

$$U_3^T\ [\dot{W}_1\ \dot{W}_2\ \dot{W}_3\ \ddot{X}_1\ \ddot{X}_2\ \ddot{X}_3]^T = U_5 \tag{31}$$

the three nonholonomic constraints are

$$\begin{aligned} &R_1^T\ A_1\ W_1 = 0 \\ &R_2^T\ [A_1^T\ A_2\ W_2 - W_1] = 0 \\ &R_3^T\ [A_2^T\ A_3\ W_3 - W_2] = 0 \end{aligned} \tag{32}$$

Differentiation of Equation (32) with respect to time and simplification renders

$$U_4^T\ [\dot{W}_1\ \dot{W}_2\ \dot{W}_3\ \ddot{X}_1\ \ddot{X}_2\ \ddot{X}_3]^T = U_6 \tag{33}$$

Let $U_7 = [U_3 \vdots U_4]^T$. Equations (30), (31), (33) may be combined to derive

$$\begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \\ \Lambda_1 \\ \Lambda_2 \\ \Lambda_3 \end{bmatrix} = \begin{bmatrix} U_7\ U_1^{-1} U_7^T \end{bmatrix}^{-1} \begin{bmatrix} (U_5^T \vdots U_6^T)^T - U_7\ U_1^{-1}\ U_2 \end{bmatrix} \tag{34}$$

This derivation could also have been carried out for $\Gamma = (\Gamma_1\ \Gamma_2\ \Gamma_3)^T$ or $\Lambda = (\Lambda_1\ \Lambda_2\ \Lambda_3)^T$.

In what follows Γ and Λ are eliminated in two successive steps and the system is reduced to a minimal 3 degree of freedom.

To eliminate Γ parametric equations are written for the accelerations. Let $U_3^T = [U_8 \vdots U_9]$. Equation (31) may be written

$$U_8\ \dot{W} + U_9 \dot{X} = U_5 \tag{35}$$

or

$$\begin{bmatrix} \dot{W} \\ \dot{X} \end{bmatrix} = \begin{bmatrix} I \\ -U_9^{-1}\ U_8 \end{bmatrix} \dot{W} + \begin{bmatrix} 0 \\ U_9^{-1} U_5 \end{bmatrix} \tag{36}$$

Let

$$\begin{bmatrix} I \\ -U_9^{-1}\ U_8 \end{bmatrix} = U_{10} \tag{37}$$

using Equations (36) and (37) in (30) renders

$$U_1 \, U_{10} \, \dot{W} + U_1 \begin{bmatrix} 0 \\ U_9^{-1} \, U_5 \end{bmatrix} = U_2 + U_3 \, \Gamma + U_4 \, \Lambda \tag{38}$$

It is shown in [31] that U_{10}^T is orthogonal to U_3. Multiplying both sides of (38) by U_{10}^T eliminates Γ.

$$(U_{10}^T \, U_1 \, U_{10}) \, \dot{W} = U_{10}^T \, [-U_1 \begin{pmatrix} 0 \\ \cdots \\ U_9^{-1} U_5 \end{pmatrix} + U_2 + U_4 \, \Lambda] \tag{39}$$

Equation (39) describes a system of dimension 18 where all translational motion is eliminated. For this system again one could derive an expression for Λ as function of state (θ, W) and inputs or further reduce the dimension by parameterizing the constrained subspace. The latter is carried on below.

The 18 state variable may be written in terms of the three physical angles ϕ_1, ϕ_2, and ϕ_3 (Figure 1). For the first element:

$$\begin{aligned} &\theta_{11} = \phi_1, \quad \theta_{12} = \theta_{13} = 0 \\ &W_{11} = \dot{\theta}_{11}, \quad W_{12} = W_{13} = 0 \end{aligned} \tag{40}$$

For body 2:

$$\begin{aligned} &\theta_{21} = \phi_1, \ \theta_{22} = \phi_2, \ \theta_{23} = 0 \\ &W_2 = [\dot{\phi}_2, \ \dot{\phi}_1 \ \sin\phi_2, \ \dot{\phi}_1 \ \cos \phi_2]^T \end{aligned} \tag{41}$$

For body 3:

$$\begin{aligned} &\theta_{31} = \phi_1, \ \theta_{32} = \phi_3, \ \theta_{33} = 0 \\ &W_3 = [\dot{\phi}_3, \ \dot{\phi}_1 \ \sin\phi_3, \ \dot{\phi}_1 \ \cos \phi_3] \end{aligned} \tag{42}$$

From the latter three equations

$$\dot{W} = U_{11} \, \ddot{\phi} + U_{12} \tag{43}$$

where U_{11} and U_{12} are given in the appendix. Substitution of Equation (43) in Equation (39) and left-multiplying both sides by U_{11}^T results in the derived equations and Λ are eliminated because

$$U_{11}^T \, U_{10}^T \, U_4 = 0 \tag{44}$$

The final equations in ϕ are:

$$(U_{11}^T \, U_{10}^T \, U_1 \, U_{10} \, U_{11}) \, \ddot{\phi} = U_{11} \, U_{10} \, [-U_1 \begin{pmatrix} 0 \\ \cdots \\ U_9^{-1} U_5 \end{pmatrix} + U_2 - U_1 \, U_{10} \, U_{12}] \tag{45}$$

For brevity equation (45) is rewritten as

$$\begin{aligned} U_{13} \; = \; & U_{14} \, [\dot{\phi}_1^2 \ \dot{\phi}_2^2 \ \dot{\phi}_3^2 \ \dot{\phi}_1\dot{\phi}_2 \ \ \dot{\phi}_1\dot{\phi}_3 \ \ \dot{\phi}_1\dot{\phi}_3 \ \ \dot{\phi}_2\dot{\phi}_3] \\ & + U_{15} \, G + U_{16} \, T \end{aligned} \tag{46}$$

where G is a vector of gravity terms and T the vector of input torques. Matrices U_{13}, U_{14}, U_{15} and U_{16} are given in the appendix. The final equations were also derived in (24) and are not repeated here.

IV. CONCLUSIONS

The state space developed here and the Newton-Euler formulation allow a modular simulation of large mechanical systems -- one microprocessor per rigid link. The inputs and the state of the whole system are provided to all modules. Based on this information, each module computes the forces of constraint that directly influence its own dynamics. These locally computed forces (and torques), plus the relevant inputs and the local state are then entered in the dynamic quations of the link to compute the next state (or the rate of change of the state with respect to time in continuous systems). Further reduction of loads is possible by allocating computation of feedback signals and constraint forces to other processors.

The division of the computations in this formal manner should allow simulation of large mechanical systems in real time. Other criteria may be used for division of such large tasks. The above modular division minimizes the amount of information that is exchanged between the processors. It also establishes a one-to-one correspondence between the microprocessors and the rigid body segments.

Since the modules are simpler and each simulates primarily the same set of equations, interfacing, programming and debugging of the modules should be easier and faster. Finally such a system is flexible to allow for simulation of a range of movements from the gross to the refined.

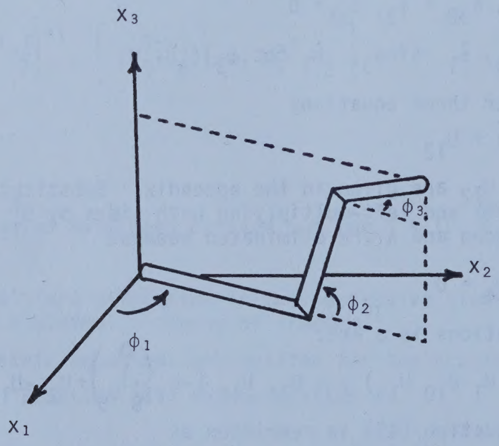


FIGURE 1 Three-Link Multi-Body System

REFERENCES

1. Bejczy, A.K., "Issues in Advanced Automation for Manipulator Control," Proc. 1976 Joint Automatic Control Conference, Purdue University, pp. 700-711.

2. Dubowsky, Steven, "On the Adaptive Control of Robotic Manipulators: The Discrete Time Case," Proc. 1981 Joint Automatic Control Conference, Charlottsville Paper TA-2B, 1981.

3. Munson, George E., "Robot Control - An Overview," Proc. 1977 Joint Automatic Control Conference, San Francisco, pp. 714-719, 1977.

4. Begeman, P.C., King, A.I., Levine, R.S., and Viano, D.C., "Biodynamic Response of the Musculoskeletal System to Impact Acceleration," Proc. Twenty-fourth Stapp Car Crash Conference, pp. 477-509, Troy, Michigan, 1980.

5. Pritz, Howard B., "Comparison of the Dynamic Responses of Anthropomorphic Test Devices and Human Anatomic Specimens in Experimental Pedestrian Impacts," Proc. Twenty-second Stapp Car Crash Conference, Ann Arbor, Michigan, 1978, pp. 341-358.

6. Wismans, J., et al., "Child Restraint Evaluation by Experimental and Mathematical Simulation," Proc. Twenty-third Stapp Car Crash Conference, San Diego, 1979, pp. 381-415.

7. Fleck, J.T., et al., "An Improved Three-Dimensional Computer Simulation of Vehicle Crash Victims," Vol. I report DOT HS-801 507, 1975.

8. Jacobsen, S.C., et al., "Performance of the Utah Prosthetic Arm," Proc. 1976 Joint Automatic Control Conference, Purdue University, pp. 727-733.

9. Leifer, L., et al., "A Robotic Aid for the Severely Disabled -- Microprocessor Controls," Proc. 1979 Joint Automatic Control Conference, Denver, pp. 777-782, 1979.

10. Roberts, W.M., Levine, W.S., and Zajac III, F.E., "Propelling a Torque Controlled Baton to a Maximum Height," IEEE Trans. Automatic Control, Vol. AC-24, No. 5, pp. 779-782, Oct. 1979.

11. Zheng, Yuan-Fang, "The Study of a Nine Link Biped Model with Two Feet," Report #TH-80-HH-058, The Ohio State University, Department of Electrical Engineering, Autumn 1980.

12. Park, K.C., "Formulation, Solution Strategies and Computer Implementation of Structural Crash Simulation, in Measurement and Prediction of Structural and Biodynamic Crash-Impact Response," Winter Annual Meeting of ASME, 1976, pp. 49-62.

13. Garrott, R., and Scott, R.A., "Improvement of Mathematical Models for Simulation of Vehicle Handling," Vol. 7, DOT-HS-805-370, National Highway Traffic Safety Administration, March 1980.

14. Kane, T.R., and Levinson, D.A., "Formulation of Equations of Motion for Complex Spacecraft, J. of Guidance and Control, Vol. 3, No. 2, pp. 99-112, 1980.

15. Gordon, Stephen L., et al., "Dynamic Characteristics of Human Leg Joints," Proc. Twenty-first Stapp Car Crash Conference, New Orleans, 1977, pp. 419-441.

16. Andriacchi, T.P., Mikosz, R.P., Hampton, S.J., and Galante, J.O., "Model Studies of the Stiffness Characteristics of the Human Knee Joint," 24th Annual Meeting Orthopaedic Research Society, Dallas, Texas, 1978.

17. Wismans, Jac, "A Three-dimensional Mathematical Model of the Human Knee Joint," Thesis, Department of Mechanical Engineering, Eindhaven University of Technology, the Netherlands, 1980.

18. Huston, J.C., and Advani, S.H., "Three Dimensional Model of the Human Head and Neck for Automobile Crashes," in Mathematical Modeling Biodynamic Response to Impact, SAE publication, SP-24, pp. 9-20, 1976.

19. Tennyson, S.A. and King, A.I., "A Biodynamic Model of the Human Spinal Column," in Mathematical Modeling Biodynamic Response to Impact, SAE publication SP-412, pp. 31-44, 1976.

20. Seireg, A., Baz, A., and Patel, D., "Supportive Forces on the Human Body During Underwater Activities," J. Biomechanics, Vol. 4, pp. 23-30, 1971.

21. Morawski, J.M., "Control Systems Approach to a Ski-Turn Analysis," J. Biomechanics, Vol. 6, pp. 267-279, 1973.

22. Kane, T.R. and Scher, M.P., "Human Self-Rotation by Means of Limb Movement," J. Biomechanics, Vol. 3, pp. 39-49, 1970.

23. Townsend, M.A., and Tsai, T.C., "Biomechanics and Modelling of Bipedal Climbing and Descending," J. Biomechanics, Vol. 9, pp. 227-239, 1976.

24. Hemami, H., Jaswa, V.C., and McGhee, R.B., "Some Alternative Formulations of Manipulator Dynamics for Computer Simulation Studies," Proc. of 13th Allerton Conference on Circuit and System Theory, University of Illinois, pp. 124-140, October, 1975.

25. Hooker, W.W., and Margulies, G., "The Dynamical Attitude Equations for an N-body Satellite," The J. of the Astronautical Sciences, Vol. XII, No. 4, pp. 123-128, Winter 1965.

26. Luh, J.T.S., Walker, N.W., and Paul, R.P.C., "On-line Computational Scheme for Mechanical Manipulators," Journal of Dynamic Systems, Measurement and Control, Vol. 102, pp. 69-76, June 1980.

27. Armstrong, W.W., "Recursive Solution to the Equations of Motion of an N-Link Manipulator," Proc. 5th World Congress on the Theory of Machines and Mechanisms, Vol. 2, pp. 1343-1346, July 1979.

28. Orin, D.E., et al., "Kinematic and Kinetic Analysis of Open-Chain Linkages Utilizing Newton-Euler Methods," Math Biosciences, 43, p. 107-130, 1979.

29. Hemami, H., and Wyman, R.F., "Modeling and Control of Constrained Dynamic Systems with Application to Biped Locomotion in the Frontal Plane," IEEE Transactions, AC, Vol. 24, No. 2, August 1979, pp. 526-535.

30. Takegaki, M., and Arimoto, S., "A New Feedback Method for Dynamic Control of Manipulators," Journal of Dynamic Systems, Measurement and Control, Vol. 103, pp. 119-125, June 1981.

31. Hemami, H., "A State Space Model for Interconnected Rigid Bodies," to appear in IEEE Transitions, AC, Vol. 27, No. 2, April, 1982.

32. Grygier, Paul A., "Distributed Simulation Models for Transportation Research Using a Computer Network," Ph.D. Dissertation, The Ohio State University.

33. Wongchaisuwat, C., "Modeling and Control of Human Knee Joint in the Sagittal Plane," M.S. Thesis, The Ohio State University, December, 1981.

34. Kahn, M.E., and Roth, B., "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains," Trans. of the ASME, J. of Dynamic Systems, Measurements, and Control, pp. 164-172, September, 1971.

EXTENSIONS OF MULTIVARIABLE CONTROL SYSTEM DESIGN TECHNIQUES TO LARGE SCALE SYSTEMS

W. S. LEVINE AND A. LEUNG
Department of Electrical Engineering
University of Maryland
College Park, Maryland 20742

ABSTRACT

There has been a great deal of excellent recent work on the "classical" design of multivariable feedback control systems. Among other contributions this work has clarified the relation between the LQG approach and frequency response based design techniques. The purpose of this paper is to extend some of these results to large scale systems. For our purposes, a system is large scale if it is not feasible to control it via a completely centralized controller.

1. INTRODUCTION

One of the central topics in modern control theory is the design of multivariable feedback controls. The search for better design methods has motivated a great deal of outstanding research into the fundamentals of feedback control and both linear and nonlinear systems. In addition, this search has lead to considerable debate over which of several very different design techniques is best. As one might expect, the best design method is rather strongly dependent on the class of problems under consideration.

In this paper we are interested in multivariable servomechanism problems subject to the constraint that the control must be decentralized. The requirement for decentralized control arises in a wide variety of practical control problems. One such example is electric power systems where the large geographical distances between generating stations makes purely centralized control impractical. Another, and perhaps surprising example, is in robot controls where, usually, each joint is independently servoed because this is cheaper than coupled control and, so far, is good enough. Note that, in both of these examples, there is some centralization of control. However, this takes place "above" or "outside" a decentralized servomechanism.

A precise definition of what we mean by a decentralized servomechanism requires some background information. We give this background information and the definition in the following section. The third section of this report contains our results. These consist of the complete extension of Mayne's Sequential Return Difference method [1] for the design of multivariable feedback controls to our class of problems. We remark that Rosenbrock's INA method has been similarly extended by Bennett and Baras [2]. This research is far from complete and, at this time, raises many questions that we have not yet answered. We conclude, in section four, with a discussion of some of these questions.

2. BACKGROUND

Generally, the techniques that have been proposed for the design of multivariable feedback controls divide into two categories. There are the so-called LQG (for linear quadratic gaussian) techniques. For our purposes, there are three comments that need to be made about LQG design techniques.

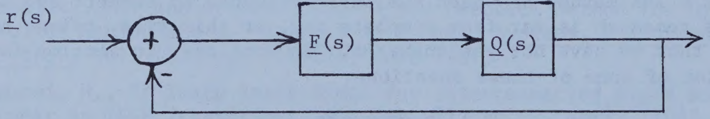
Acknowledgement: This research was supported, in part, by the U. S. Dept. of Energy under contract DE-AC01-80RA50420 and, in part, by the Computer Science Center of the University of Maryland.

The first is that the designer chooses the weighting matrices in a quadratic performance measure and, possibly, some noise convariance matrices and then the design algorithm gives the control (see Athans [3] or, for a very up to date description, any of Doyle and Stein [4], Safanov et a [5], Cruz et al [6] or Lehtomaki et al [7]). Secondly, LQG designs tend to be very robust, that is, insensitive to disturbances. Thirdly, the LQG design methods automatically produced completely centralized controls. In fact, the attempt to apply LQG methods to decentralized control problems lead to the famous Witsenhausen counter-example [8] about which we will say more shortly.

In comparison to classical feedback controls, LQG designs tend to be very complicated and expensive. Thus, several people attempted to find methods for designing multivariable feedback controls that would be more like classical feedback controls. The result is a series of design methods that are usually referred to as frequency response design methods. In particular, these methods include Rosenbrock's INA method [9], Mayne's Sequential Return Difference method [1] and MacFarlane's Characteristic Locus method [10]. Again, three comments are in order. First, all of these methods produce controllers that are known to be less robust than those designed via LQG techniques. Thus, in many situations LQG designs are better than frequency response designs. Second, the advantage of frequency response design methods is that the control system designer retains much more control over the details of his design. Of most relevance to our problem is the fact that he is often able to force his design to be decentralized. Third, Zames [11] has recently demonstrated a way to produce frequency based designs that are robust.

Finally, we conclude this very brief survey of design methods for multivariable control systems by mentioning two papers that bridge the gap between LQG and frequency response design methods. These are by Youla et al [12], [13] and describe an LQG design done in the frequency domain.

We have already given some hints about what happens when the controller is not, for one reason or another, completely centralized. Here, we want to make this discussion precise. First, we are only interested in linear servomechanisms. That is, we assume that the given system is linear and, in fact, time invariant and that the goal of the feedback controller is to cause the system output to match an externally specified set point. However, we do not require our plant transmission function to be square. Second, we assume that the given plant has multiple inputs and multiple outputs. Thus, the feedback controller is some mapping of outputs into inputs. Since we are dealing with linear time invariant systems and linear time invariant feedback controllers we can specify our controller as a matrix transfer function (see Fig. 1 for notation).



where $\underline{r}(s)$= set point (an m-vector)
$\underline{y}(s)$= output (an m-vector)
$\underline{Q}(s)$= plant transfer function (nxm)
$\underline{F}(s)$= controller (mxn) (to be designed)

Fig. 1

Finally, a decentralized controller is a "block diagonal" $\underline{F}(s)$. For example

$$\underline{F}(s) = \begin{bmatrix} \underline{F}_1(s) & \underline{0} \\ \underline{0} & \underline{F}_2(s) \end{bmatrix} \tag{1}$$

where neither $\underline{F}_1(s)$ nor $\underline{F}_2(s)$ has to be square.

Two points need to be emphasized. First, the Witsenhausen counter example shows that such a block diagonal feedback controller is, in general, not optimal in the LQG sense. Furthermore, the LQG optimum is not necessarily even linear. Finally, imposing the block diagonal constraint on an LQG approach can lead to "signalling". That is, the use of the control signal to convey information about observations through the plant. This is most undesireable for servomechanisms.

Second, it is implicit in our problem formulation that arbitrary changes of coordinates are not allowed, either in the input or the output vector space. We do, however, allow only block diagonal changes of coordinates. That is, $\underline{Q}(s)$ and $\hat{\underline{Q}}(s)$ are equivalent if and only if there exists $\underline{P}$ and $\underline{T}$ such that

$$\hat{\underline{Q}}(s) = \underline{P}\ \underline{Q}(s)\underline{T} \tag{2}$$

where $\underline{P}$ is nxn, invertible and of the form $\underline{P} = \begin{bmatrix} \underline{P}_1 & \underline{0} \\ 0 & \underline{P}_2 \end{bmatrix}$

$\underline{T}$ is mxm, invertible and of the form $\underline{T} = \begin{bmatrix} \underline{T}_1 & \underline{0} \\ \underline{0} & \underline{T}_2 \end{bmatrix}$

The physical meaning of this should be apparent from our problem formulation. Arbitrary changes of coordinates require central access to all components of $\underline{y}$ or $\underline{u}$. This is not allowed.

To summarize, the problem of interest to us is not amenable to a pure LQG approach. In the following section we describe a frequency response based design method that is likely to be useful for this class of problems.

3. RESULTS

There are several ways to use frequency response data to design decentralized multivariable feedback controllers. Bennett and Baras [2] give one such method. Here we present a straightforward extension of Mayne's Sequential Return Difference method from diagonal feedback to block diagonal feedback.

The result is in two parts. We first prove a stability theorem based on a sequence of partial return differences. Secondly, we describe a design procedure based on the stability theorem. We first establish some notation (see Fig. 1.)

$\underline{Q}(s)$ is an nxm transfer function

$$\underline{F}(s) = \text{block diagonal}\ (\underline{F}_1(s), \underline{F}_2(s) \ldots . \underline{F}_\ell(s)) \tag{3}$$

where $F_i(s)$ is $n_i x m_i$ and $\sum_{i=1}^{\ell} n_i = n$, $\sum_{i=1}^{\ell} m_i = m$

$$\text{Define}\quad \underline{T}(s) = \underline{I}_n + \underline{Q}(s)\underline{F}(s) \tag{4}$$

and $\underline{R}(s) = \underline{T}^{-1}(s)\,\underline{Q}(s)\,\underline{F}(s)$ (5)

Thus, $\underline{T}(s)$ is the return difference and $\underline{R}(s)$ is the closed-loop transfer function.

Define

$$\underline{F}^{i}(s) = \text{block diagonal}(\underline{F}_1(s), \underline{F}_2(s)_1 \ldots \underline{F}_i(s), \underline{0}, \underline{0} \ldots \underline{0}) \quad (6)$$

$$\underline{T}_i(s) = \underline{I}_n + \underline{Q}(s)\underline{F}^{i}(s) \quad (7)$$

$$\underline{Q}^{i}(s) = \underline{T}_i^{-1}(s)\underline{Q}(s) \quad (8)$$

$$\mathcal{T}_i(s) = \underline{I}_{n_i} + \underline{Q}_{ii}^{i-1}(s)\,\underline{F}_i(s) \quad (9)$$

where $\underline{Q}_{ii}^{i-1}(s)$ denotes the ii$^{\underline{th}}$ block of $\underline{Q}^{i-1}(s)$.

It is then easy to check that

$$\underline{T}_0(s) = \underline{I}_n \qquad \underline{Q}^{0}(s) = \underline{Q}(s)$$

$$\underline{T}_{\ell}(s) = \underline{T}(s) \qquad \underline{G}^{\ell}(s)\,\underline{F}(s) = \underline{R}(s)$$

In addition, we state without proof the following well-known stability theorem.

Theorem 1: $\det T(s) = \rho_c(s)/\rho_0(s)$ where $\rho_0(s)$ and $\rho_c(s)$ are the open and closed-loop characteristic polynomials.

The crucial result is the following:

Theorem 2: $\det \underline{T}_i(s) = \overset{i}{\underset{j=1}{\pi}} \det \mathcal{T}_i(s)$ (10)

Proof: (where we illustrate the steps with ℓ=3 and i=2) $\underline{T}_0(s) = \underline{I}_n$

$$\underline{T}_i(s) = \underline{I}_n + \underline{Q}(s)\,\underline{F}^{i}(s)$$

$$= \underline{I}_n + \begin{bmatrix} \underline{Q}_{11} & \underline{Q}_{12} & \underline{Q}_{13} \\ \underline{Q}_{21} & \underline{Q}_{22} & \underline{Q}_{23} \\ \underline{Q}_{31} & \underline{Q}_{32} & \underline{Q}_{33} \end{bmatrix} \begin{bmatrix} \underline{F}_1 & \underline{0} & \underline{0} \\ \underline{0} & \underline{F}_2 & 0 \\ \underline{0} & \underline{0} & \underline{0} \end{bmatrix}$$

$$= \underline{I}_n + \begin{bmatrix} \underline{Q}_{11}\underline{F}_1 & \underline{Q}_{12}\underline{F}_2 & \underline{0} \\ \underline{Q}_{21}\underline{F}_1 & \underline{Q}_{22}\underline{F}_2 & \underline{0} \\ \underline{Q}_{31}\underline{F}_1 & \underline{Q}_{32}\underline{F}_2 & \underline{0} \end{bmatrix}$$

$$= \underline{T}_{i-1}(s) + \begin{bmatrix} \underline{0} & \underline{Q}_{12}\underline{F}_2 & \underline{0} \\ \underline{0} & \underline{Q}_{22}\underline{F}_2 & 0 \\ \underline{0} & \underline{Q}_{32}\underline{F}_2 & \underline{0} \end{bmatrix}$$

$$= \underline{T}_{i-1}(s) + \begin{bmatrix} \underline{Q}_{12} \\ \underline{Q}_{22} \\ \underline{Q}_{32} \end{bmatrix} \begin{bmatrix} \underline{0} & \underline{F}_2 & \underline{0} \end{bmatrix}$$

$$\therefore \underline{T}_i(s) = \underline{T}_{i-1}(s) + \underline{Q}_{.i}\,\underline{F}_{i.}$$

Since $\underline{Q}^{i-1}(s) = \underline{T}^{-1}_{i-1}(s)\underline{Q}(s)$, $\underline{Q}^{i-1}_{.i}(s) = \underline{T}^{-1}_{i-1}(s)\,\underline{Q}_{.i}(s)$

Thus, $\underline{T}_i(s) = \underline{T}_{i-1}(s)\,(\underline{I}_n + \underline{Q}^{i-1}_{.i}\,\underline{F}_{i.})$

However, $\det\,(I_n + Q^{i-1}_{.i}\,F_{i.}) = \det \begin{bmatrix} \underline{I}_{m_1} & \underline{Q}^{i-1}_{12}\,\underline{F}_2 & \underline{0} \\ \underline{0} & \underline{I}_{m_2} + \underline{Q}^{i-1}_{22}\,\underline{F}_2) & \underline{0} \\ \underline{0} & \underline{Q}^{i-1}_{32}\,\underline{F}_2 & \underline{I}_{m_s} \end{bmatrix}$

$$= \det \begin{bmatrix} \underline{I}_{m_1} & \underline{Q}^{i-1}_{12}\,\underline{F}_2 \\ \underline{0} & (\underline{I}_{m_2} + \underline{Q}^{i-1}_{22}\,\underline{F}_2) \end{bmatrix} = \det\,(\underline{I}_{m_2} + \underline{Q}^{i-1}_{ii}\,\underline{F}_i)$$

$$= \det \underline{\mathcal{T}}_i(s) \qquad \underline{\text{QED}}$$

Corollary 1: $\det \underline{T}(s) = \prod_{j=1}^{\ell} \det \mathcal{T}_j(s)$

Now, if we let D denote the standard contour used in evaluating the Nyquist criterion then the standard Nyquist theorem gives us

Theorem 3: Let $\det \underline{\mathcal{T}}_i(s)$ map D into the Nyquist locus γ_i which encircles the orgin N_i times clockwise for $i=1,2,\ldots,\ell$. Then the closed loop system is asympotically stable if and only if $\sum_{i=1}^{\ell} N_i = -P_0$ where P_0 is the number of open-loop unstable poles.

Theorem 3 leads to the following procedure for designing decentralized controllers.

Step 1: Let $\underline{Q}^0(s) = \underline{Q}(s)$ and $i = 1$

Step 2: Compute $\underline{\mathcal{T}}_i(s) = \underline{I}_{m_i} + \underline{Q}^{i-1}_{ii}(s)\,\underline{F}_i(s)$

Here $\underline{F}_i(s)$ is chosen so that $\underline{\mathcal{T}}_i(s)$ is "good".

By this we mean that we have made suitable progress towards overall stability and that $\underline{\mathcal{T}}_i(s)$ is an appropriate return difference for the problem at hand. Thus, there is a sub-design problem here that could be handled by any of the standard centralized feedback design procedures, including LQG. In fact, because of its low sensitivity to disturbance, LQG would be a particularly good choice.

Step 3: If $i = \ell$, stop. Otherwise

(i) compute $\underline{\mathcal{T}}_i^{-1}(s)$

(ii) set $\underline{\bar{F}}_i(s) = \underline{F}_i(s)\,\underline{\mathcal{T}}_i^{-1}(s)$

(iii) compute $\underline{Q}^i = \underline{Q}^{i-1} - \underline{Q}^{i-1}_{\cdot i}\,\underline{\bar{F}}_i\,\underline{Q}^{i-1}_{i}$.

(iv) set $i = i+1$ and return to Step 2.

There are several open questions associated with this design procedure. One of the most obvious is how to order the subsystems (blocks). The following corollary to Theorem 3 is a partial result in this direction.

Corollary 2: If $\sum_{i=1}^{k} N_i = -P_0$ and $N_i = 0$ for $i=k+1,\dots,\ell$ then the closed loop system will still be asymptotically stable if any subset of loops $k+1,\dots,\ell$ is opened provided the subset includes all loops with indices greater than the lowest one in the set.

4. DISCUSSION

As we said earlier, there are still many unsolved questions associated with the given design procedure. Here, we briefly describe four of these questions which we believe are particularly crucial. We have already mentioned the first. How do you order the blocks? Corollary 2 shows that certain orderings are better than others but does not give any method for finding the best one. In fact, this question of ordering seems to be closely tied to all of the other questions we raise here.

We have also previously given a partial answer to our second question. How do you design the individual blocks? Although it is fairly obvious that almost any of the existing algorithms for designing multivariable feedback controls can be used effectively by a good designer it is not clear how to specify the goals for the design of the individual blocks. Thus, one cannot specify a best design procedure.

The third question deals with a very serious problem associated with the technique. Sometimes, the design procedure produces, at intermediate steps, right half plane zeros that must be dealt with in the design of individual blocks. These zeros greatly complicate the design of the individual blocks in which they appear. We are left with the following question. What causes these right half plane zeros and how can we best deal with them?

The fourth question deals with a peculiar anomaly of the method. The apparent order of the i^{th} sybsystem is sometimes much larger than the order of the overall system. This poses the following question. Why does the intermediate system order get so large and how can it be reduced? A partial answer to this question can be to approximate the intermediate system by a lower order system. This can be quite dangerous in stabilization problems. Thus, we would like a more complete answer.

In conclusion, the work described here is "in progress" and by no means completed. We believe the problem formulation accurately describes a significant class of practical problems. We also believe the problem formulation, our preliminary results and related results by others raise some important questions about feedback control. We hope to be able to answer some of these questions in the reasonably near future.

5. REFERENCES

1. Mayne, D. Q.; "The Design of Linear Multivariable Systems", Automatica, March 1973 pp. 201-207.
2. Bennett, W. and J. S. Baras, "Block Diagonal Dominance and Design of Decentralized Compensators", Proc. of Conf. on Inf. Sci and Systems, Johns Hopkins Univ., Baltimore, Md., 1979.
3. Athans, M., "The Role and Use of the Stochastic Linear Quadratic Gaussian Problem in Control System Design", IEEE Trans. on AC, Vol. AC-16, Dec. 1971, pp. 529-552.
4. Doyle, J. C. and G. Stein, "Multivariable Feedback Design:Concepts for a Classical/Modern Synthesis", IEEE Trans. on AC, Vol. AC-26, No. 1, Feb. 1981, pp. 4-16.
5. Safanov, M. G., A. J. Laub and G. L. Hartmann, "Feedback Properties of Multivariable Systems:The Role and Use of the Return Difference Matrix", IEEE Trans on AC, Vol. AC-20, No. 1, Feb. 1981 pp. 47-65.
6. Cruz, J. B., J. S. Freudenberg and D. P. Looze, "A Relationship Between Sensitivity and Stability of Multivariable Feedback Systems", IEEE Trans. on AC, Vol. AC-26, No. 1, Feb. 1981, pp. 66-74.
7. Lehtomaki, N. A., N. R. Sandell Jr. and M. Athans, "Robustness Results in Linear-Quadratic Gaussian Based Multivariable Control Designs", IEEE Trans. on AC, Vol. AC-26, No. 1, Feb. 1981, pp. 75-92.
8. Witsenhausen, H. S., "A Counter example in Stochastic Optimal Control", SIAM J. Control, Vol. 6, No. 1, 1968.
9. Rosenbrock, H. H., Computer Aided Control System Design New York, Academic, 1974.
10. McFarlane, A. G. J. and B. Kouvaritakis, "A Design Technique for Linear Multivariable Feedback Systems", Int. J. Control, Vol. 25, 1977, pp. 837-879.
11. Zames, G., "Feedback and Optimal Sensitivity:Model Reference Transformations, Multiplicative Seminorms and Approximate Inverses", IEEE Trans. on AC, Vol. AC-26, No. 2, April 1981, pp. 301-320.
12. Youla, D. C., J. J. Bongiorno Jr. and H. A. Jabr, "Modern Wiener-Hopf Design of Optimal Controllers-Part I: The Single Input-Output Case", IEEE Trans. on AC, Vol. AC-21, Feb. 1976, pp. 3-13.
13. Youla, D. C., H. A. Jabr and J. J. Bongiorno Jr., "Modern Wiener-Hopf Design of Optimal Controllers-Part II: The Multivariable Case", IEEE Trans on AC, Vol. AC-21, June 1976, pp. 319-338.

DECENTRALIZED REGULATION OF CELLULAR STRUCTURES

ÜMİT ÖZGÜNER[†]
Department of Electrical Engineering
The Ohio State University
Columbus, Ohio 43210

BÜLENT YILDRIM
Golcuk Naval Base, Turkey

ABSTRACT

A cellular structure of interconnected systems forming a two dimensional array is considered. The bi-lateral two-dimensional z-transform is employed to obtain a compact formulation for an infinite number of subsystems. The decentralized regulation of such cellular structures is formulated. The solutions may be used in modeling biological systems, traffic networks and power systems.

INTRODUCTION

A number of large-scale systems made up of an interconnection of subsystems show a regular cellular structure. Various results have recently been reported concerning single dimensional arrays of countably-infinite number of subsystems [1, 2, 3]. We would like to, in this paper, discuss the possible extention of such results to two-dimensional arrays. This will extend the applicability of the results to a more general class of large scale systems. In fact one example will be provided in the sequel.

We have to point out that even if the large scale composite system one is interested in, is comprized of a finite number of subsystems, and even if each subsystem is not precisely equal to another, the approach presented may be of value, as presenting an initial compact framework on which to base a first analysis and design.

Thus, we claim that, a number of models in traffic networks, power systems and biological systems may be imbedded in the given framework

II. Main Result

A number of recent investigations have been reported about one-dimensional cellular structures [1, 2, 3]. Following is the general representation:

$$\dot{x}_k = \sum_{i=-\infty}^{\infty} (A_{k-i}\, x_i + B_{k-i} u_i) \quad -\infty<k<\infty \tag{1}$$

Note that this implies interaction between every subsystem within the structure. A "next-neighbor affect" with unidirectional interaction and local inputs would be modeled as:

$$\dot{x}_k = A_o\, x_k + A_1\, x_{k-1} + B_o\, u_k \quad -\infty<k<\infty \tag{2}$$

[†] The first author was supported in part by NSF Grant No. ECS-7824440.

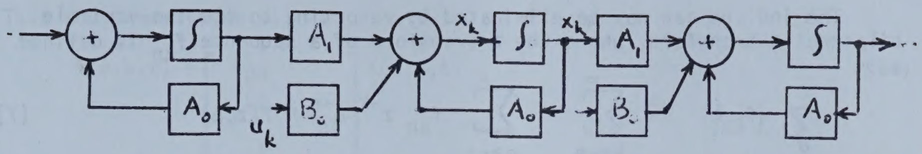


Figure 1. A one-dimensional structure with unidirectional interaction.

To eliminate the indices of the subsystems the following bilateral z-transformation of any sequence f_k is defined:

$$\mathcal{Z}(f_k) \overset{\Delta}{=} \sum_{k=-\infty}^{\infty} f_k z^{-k} \overset{\Delta}{=} F(z) \tag{3}$$

where z is a complex variable. The reverse transform is

$$f_k \overset{\Delta}{=} \left(\frac{1}{2\Pi j}\right) \oint F(z)\, z^{k-1}\, dz \tag{4}$$

A two dimensional cellular structure can also be envisaged. A generalization of (1) to two dimensions would be,

$$\dot{x}_{kn} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} (A_{k-i,n-j} X_{ij} + B_{k-i,n-j} U_{ij}) \tag{5}$$

The system with next-neighbor bi-directional interaction with local inputs would be,

$$\begin{aligned} \dot{x}_{kn} &= A_{oo}X_{kn} + A_{o1} X_{kn-1} + A_{0-1} X_{kn+1} + A_{10} X_{k-1n} \\ &\quad + A_{-10} X_{k+n} + B_{oo} U_{kn} \quad -\infty<k,\ n<\infty \end{aligned} \tag{6}$$

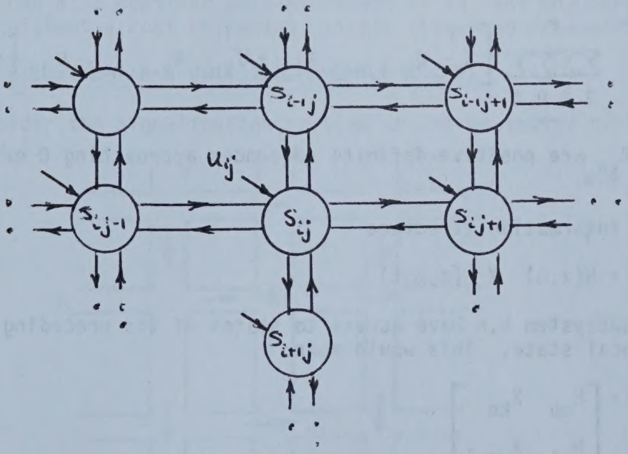


Figure 2. A two-dimensional interconnected system.

The indices can now be eliminated by resorting to the two-variable bilateral z-transform, where the z-transform of a sequence f_{kn} is defined as:

$$\mathcal{Z}\{f_{kn}\} \triangleq \sum_{k=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f_{kn}\, z^{-k}\, \omega^{-n} \triangleq F(z,\omega) \tag{7}$$

where both z and ω are complex variables. The inverse transformation is defined as:

$$f_{kn} \triangleq \left(\frac{1}{2\Pi j}\right)^2 \oint \oint F(z,\omega)\, z^{k-1}\, \omega^{n-1}\, dz\, d\omega$$

where the path of intergration usually taken is the unit circle for both of the variables.

One would use convolution to transform the general systems (1) and (5) as

$$\dot{x}(z,t) = A(z)\, X(z,t) + B(z)\, U(z,t)$$

and

$$\dot{x}(z,\omega,t) = A(z,\omega)\, X(z,\omega,t) + B\,(z,\omega)\; U\,(z,\omega,t) \tag{8}$$

respectively. The simplified systems (2) and (6) can be written as:

$$\dot{x}(z,t) = (A_o + z^{-1} A_1)\; X\,(z,t) + B_o\; U\,(z,t) \tag{9}$$

and

$$\dot{x}(z,\omega,t) = (A_{oo} + \omega^{-1} A_{01} + \omega A_{o-1} + z^{-1} A_{10} + z A_{-10})\; X\,(z,\omega,t) + B_{oo}\;(z,\omega,t) \tag{10}$$

Regulation and Decentralized Feedback.

Extention of results in the one dimensional case [2] to two dimensions is fairly straight forward. We can define a quadratic peformance index as:

$$J = \frac{1}{2} \int_o^{\infty} \sum_{j}\sum_{k}\sum_{n}\sum_{i}{}_{-\infty}^{\infty} \left[X_{kn}^T\, Q_{k-i,n-j}\, X_{ij} + U_{kn}^T\, R_{k-i,n-j}\, U_{ij} \right] dt \tag{11}$$

where Q_{kn} and R_{kn} are positive-definite sequences approaching 0 exponentially as $k,n \to \pm\infty$.

Assume an information structure

$$Y(z,\omega,t) = H(z,\omega)\;\; X\;\; (z,\omega,t) \tag{12}$$

Example: Let subsystem k,n have access to states of its preceding neighbors and its local state. This would mean:

$$y_{kn} = \begin{bmatrix} H_{oo} & X_{kn} \\ H_{o1} & X_{kn-1} \\ H_{10} & X_{k-1n} \end{bmatrix} \tag{13}$$

This would transform to

$$Y(z,\omega,t) = \begin{bmatrix} H_{oo} & \\ w^{-1} & H_{o1} \\ z^{-1} & H_{10} \end{bmatrix} X(z,\omega,t) \tag{14}$$

Either way note that information from other subsystems is kept separate from local information.

We would like to find the best linear feedback F under the given information structure.

Define: $\hat{A}(z,\omega) \triangleq A(z,\omega) + B(z,\omega)\ FH(z,\omega)$ (15)

$\hat{Q}(z,\omega) \triangleq Q(z,\omega) + H(z^{-1},\omega^{-1})^T F^T R(z,\omega) H(z,\omega)$ (16)

Let $< \ >_\infty$ denote the {00}'th term of the inverse transform.

Theorem 1

The optimal F satisfies

$$< R(z,\omega) FH(z,\omega)\ L(z,\omega)\ H^T(z^{-1},\omega^{-1}) >_\infty + < B(z^{-1},\omega^{-1})^T\ K(z,\omega)\ L(z,\omega)\ H^T(z^{-1},\omega^{-1}) >_\infty = 0 \tag{17}$$

where L and K are solutions of

$$L(z,\omega)\ \hat{A}(z^{-1}, \omega^{-1})^T + \hat{A}(z,\omega)\ L(z,\omega) + I = 0 \tag{18}$$

$$K(z,\omega)\ \hat{A}(z,\omega) + \hat{A}(z^{-1}, \omega^{-1})^T\ K(z,\omega) + \hat{Q}(z,\omega) = 0 \tag{19}$$

No claim is made about the ease of solution of the above equations except in simple examples.

One can also consider pole-placement or in fact any servomechanism problem, without a cost criterion, within the above framework.

III. Signalization in a Cellular Traffic Network

Consider the signalization problem in the congested traffic network of Figure 3.

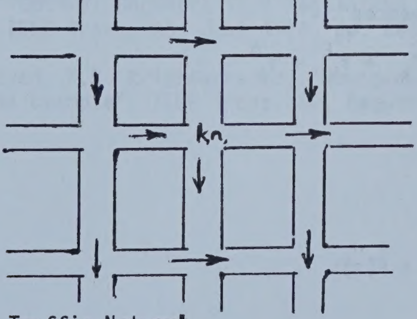


Figure 3. A Traffic Network.

It has been assumed, for simplicity, that this network is made up of one-way streets with no-turn intersections. The dynamics at intersection k,n can be written as:

$$\dot{q}^E_{kn} = t^E_{kn} - S_{kn}U^E_{kn} + f^E_{kn} \qquad (20)$$

$$\dot{q}^N_{kn} = t^N_{kn} - S_{kn}\,(1 - U^E_{kn}) + F^N_{kn} \qquad (21)$$

where the index E indicates the East-West direction, N indicates the North-South direction and furthermore,

q: queue length at traffic light (cars)

t: incoming traffic from preceding intersection (cars/sec)

f: incoming traffic from outside network, assumed constant (cars,sec)

u: signal split (given as percentage of cycle length)

s: throughput of intersection (cars,sec)

Note that if s is taken to be a constant, there is only one control variable u^E as the cycle length has been assumed to be constant, and the yellow signal duration has been ignored.

A further approximation would be assuming an immediate transfer of traffic from one intersection to another by equating t^E to the su^E of the preceding intersection.

Let us disucss the one-dimensional case. Assume s to be the same for all intersections. The model is now,

$$\dot{q}^E_k = t^E_k - su^E_k + f^E_k \qquad (22)$$

$$\dot{q}^N_k = f^N_k - S(1 - u^S_k) \qquad (23)$$

and $t^E_k = su^E_{k-1}$ (24)

There are now two options:

The two equations above can be reduced to one by defining the aggregate queue lengths and "distrubances".

$$\hat{q}_k = q^E_k + q^N_k \;, \quad \hat{f}_k = f^E_k + f^N_k - s \qquad (25)$$

to give

$$\dot{\hat{q}}_k = su_{k-1} + \hat{f}_k \qquad (26)$$

In the z-domain we have,

$$\dot{\hat{Q}}(z,t) = Sz-1\; U(z,t) + \hat{F}(z) \qquad (27)$$

One can now assume that the steady-state solution is given and regulate for variations around it. This is somewhat similar to an approach in practice of down-loading (say two or three times a day) the signal split from a central computer and letting local signal controllers adjust slightly according to local measurements.

This would be a reasonable approach from the global viewpoint. The second option is considering the logic affecting a local controller. This might dictate regulating the difference between queue lengths, i.e. define

$$\dot{\bar{q}}_{,} = q_n^N - q_{,}^E \quad , \quad \bar{f}_k = f_k^N - f_k^E - S \tag{28}$$

to give

$$\dot{\bar{q}}_k = 2\ su_k - su_{k-1} + \bar{f}_k \tag{29}$$

which in the z domain transforms to

$$\dot{\bar{Q}}(z,t) = s(2s-z^{-1})\ U\ (z,t) + \bar{F}(z) \tag{30}$$

One may now consider a PI controller as more in line with the general philosophy of the above model.

All the above models can be extended to two dimensions in a fairly straight-forward way.

IV. Conclusion

A model for two dimensional cellular structures has been considered. It has been shown that the two-dimensional z-transform provides a compact formulation for analysis and design of feedback controllers.

A number of models in biological systems, power and traffic systems may be considered within this framework.

REFERENCES

[1] S.M. Melzer, B.C. Kuo, "Optimal Regulation of Systems Described by a Countably Infinite Number of Objects", Automatica, Vol. 7, pp. 359-366, 1971.

[2] K-C. Chu, "Optimal Decentralized Regulation for a String of Coupled Systems", IEEE Trans. AC. June 1974, pp. 243-246.

[3] M.L. El-Sayed, P.S. Krishnaprasad, "Homogeneous Interconnected Systems: An Example", IEEE Trans. AC, August 1981, pp. 894-901.

NOISE REJECTION IN ARRAY DATA

M. SIMAAN
Department of Electrical Engineering
University of Pittsburgh
Pittsburgh, PA 15261

ABSTRACT

Linear arrays of sensors are used for measurements of signals in a variety of applications such as biomedical, nondestructive testing, seismic and radar. Typically the signal to be measured travels along the array at a given velocity while other interfering signals, referred to as noise, also travel along the array at a different velocity. The purpose of this paper is to discuss a new processing method for suppressing the noise without affecting the desired signal. This method involves the design of special filters which are applied, off-line, to the array data. The output data, after filtering, will consist of the input data from which the noise has been optimally suppressed.

INTRODUCTION

In many applications; such as radar, seismic and biomedical; measurements of signals are often made by arrays of sensors [1]. Typically, the signal to be measured is a bandlimited pulse, or series of pulses, which travel along the array at a given velocity. Often, other signals, which will be referred to as noise also travel along the array at a different velocity and interfere with the signals that are being measured. The two dimensional array of data obtained by such measurements, therefore, consists of a superposition of a desired signal and an undesired noise. If the frequency spectra of the signal and noise overlap, band pass filtering to suppress the noise will result in loss of the signal frequencies that overlap with noise frequency band.

---

*This work was supported in part by a grant from Gulf Science and Technology Company.

Typically, the 2-dimensional spectrum of the data, under consideration will look as shown in figure 1. The signal and the noise are travelling at different velocities and as a result they appear separated in the spectrum making different angles with the temporal frequency axis in the spectrum. If the noise velocity is low, spacial aliasing will occur and as a result, 2-dimensional velocity filtering, which essentially removes the frequencies at a given velocity in the spectrum, becomes infeasible to do. In this paper, we describe, a procedure for optimally attenuating the noise without affecting the desired signal.

OPTIMUM ARRAY FILTERS

Consider a linear array of N equispaced sensors as shown in figure (2). Let D be the spacing between two consecutive sensors. Let V_d be the velocity of the desired signal whose plane wave makes an angle θ_d with the array and let V_u be the velocity of an undesired signal (noise) whose plane wave makes an angle θ_u with the array. If the sensors are sensitive to the horizontal component of the two signals; then after measurements; the two dimensional array data can be shifted to align along the desired signal. The apparent travel time of the noise signal between two consecutive sensors can be computed as:

$$\tau = D\left(\frac{\cos \theta_u}{V_u} - \frac{\cos \theta_d}{V_d} \right) \tag{1}$$

Let $F_i(f)$, i=1, . . .N be the frequency spectra of filters to be applied to the shifted outputs of the sensors. If after applying the filters, the data is averaged over all sensor, the reject response of the filters will be:

$$R(f) = \sum_{i=1}^{N} e^{j2\pi f\tau(i-1)} F_i(f) . \tag{2}$$

The filters $F_i(f)$ are to be designed based on two criteria:

a) To preserve the desired signal. For this reason, we will impose the constraint

$$\sum_{i=1}^{N} F_i(f) = 1 \qquad (3)$$

on the filter coefficients [2], and

b) To attenuate as much as possible the undesired signal. For this reason, we will minimize the criterion function:

$$E = \sum_{i=1}^{N} \left| R(f) - R_i(f) \right|^2 \qquad (4)$$

where $R_i(f)$ is the i^{th} component in the summation in (1).

This constraint minimization of (4) subject to (3) can be done analytically and closed-form expressions for the filter coefficients in the frequency domain can be derived. The output waveform after alignment and filtering of the sensors outputs will consist of the desired signal with minimum interfering noise.

As an example, figure (3a) illustrates the effect of this procedure for an 8-sensor array. The sensor spacing is 25 ft and the time separation of the noise is τ=5 msec between two consecutive sensors. The desired signal is 60 dB in power below the noise and consist of three bandlimited pulses at t=0, 100 and 150 m sec. After applying the optimum filters the output is shown in figure (3b). Clearly the desired signal has been recovered effectively.

DATA PROCESSING SCHEME

The main drawback of the method discussed above is that the output consists of one single waveform. Often, in many applications, the desired output is required to consist of as many channels as the number of sensors on the array. That is, the processing scheme, which takes N waveforms as input data, should produce also N waveforms as output data in which the noise has been suppressed. The approach, considered in this paper, produces an output which has slightly less than N waveforms. In order to achieve this, we will take advantage of an important property of the optimum filters discussed earlier. It is possible to show that with the optimum filters the degree of attenuation of noise that can be otained with an array that has a small number of sensors is quite significant. For instance an average attenuation of about 30 dB can be achieved with an array of three sensors. This quite sufficient in a large number of applications.

Our approach, therefore is to consider a sliding aperture (figure 4) along the array of M sensors (typically three to five). Each aperture is treated as a sub-array for which optimum sub-array filters are designed and applied as discussed in the previous section. The only information needed to design these filters are the time delays of the noise relative to the aligned desired signal in the aperture. After these filters are designed and applied, an output waveform is obtained and is used to replace the output of the sensor at the left most edge of the aperture. When this is completed the aperture is moved one sensor to the right and the procedure repeated for this new aperture. We note that if the velocities of signal and noise do not charge across the array; the same filters can be used over again; otherwise a new set of filters will have to be redesigned. When the right most edge of the array is reached, a total of N-M+1 output waveforms will have been generated. This two-dimensional data will therefore consist of the sensor output waveforms from which the noise has been suppressed.

In order to illustrate the effectiveness of this procedure, consider the 48-sensor array data shown in figure 5a. The sensors spacing is 20 ft and the data has been aligned along the desired signal. The apparent velocity of the noise is 400 ft/sec. resulting in a noise travel time of 50 msec between two consecutive sensors. The desired signal pulse is bandlimited between 4 and 60 Hz and the noise pulse is bandlimited between 4 and 25 Hz. The average noise magnitude is five times (14 dB) larger than the average signal magnitude. After optimum filtering with an aperture of 5 sensors, the output data which consists of 44 waveforms is shown in figure 5b. Clearly the noise has been significantly suppressed. The residual noise which is concentrated at the alias frequency of the filters (20 Hz), is about 10 dB in magnitude below the signal resulting in about 24 dB attenuation. Other noise frequencies have been attenuated even further.

CONCLUSIONS

In this paper, we have discussed a new procedure for suppressing linear noise in array data. The procedure is based on a sliding sub-array aperture, and involves optimally filtering the noise in each aperture. The output data consists of the input array data from which the noise has been suppressed and the desired signal preserved with little or no distortion.

REFERENCES

(1) Dudgeon, D.E., "Fundamentals of Digital Array Processing," Proc. of the IEEE, Vol. 65, No. 6, June 1977, pp. 898-904.

(2) Levin, M.J., "Maximum Likelihood Array Processing," Report #31, MIT Lincoln Laboratory, December 1964.

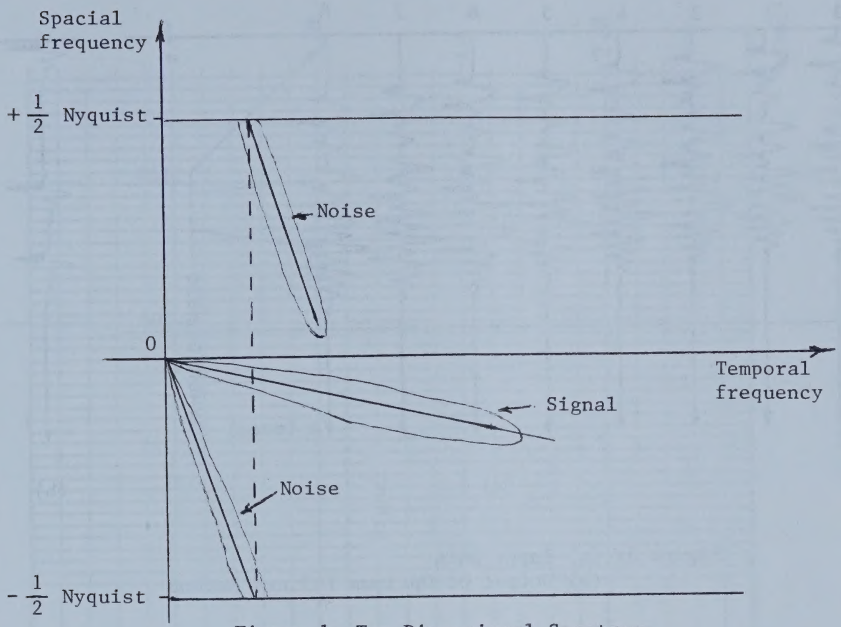


Figure 1: Two Dimensional Spectrum

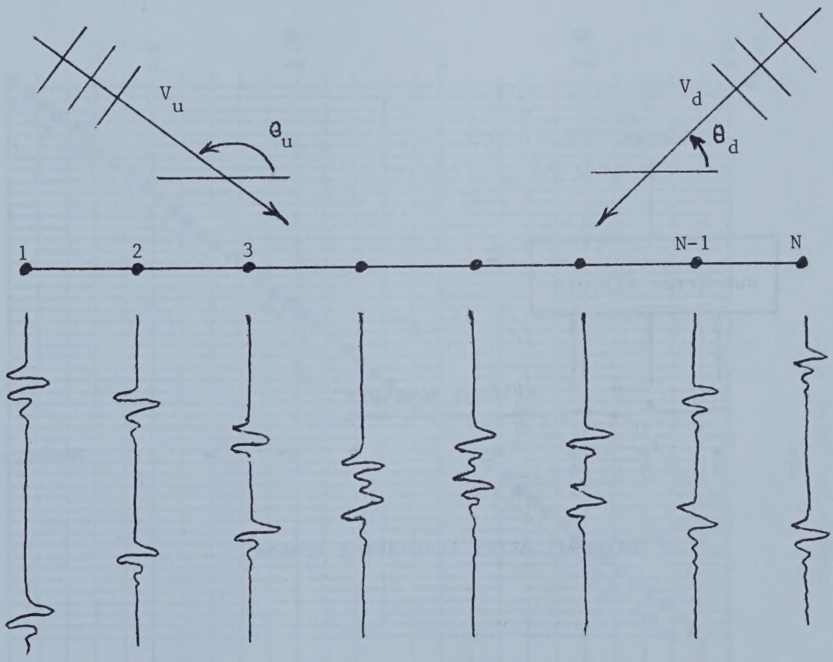
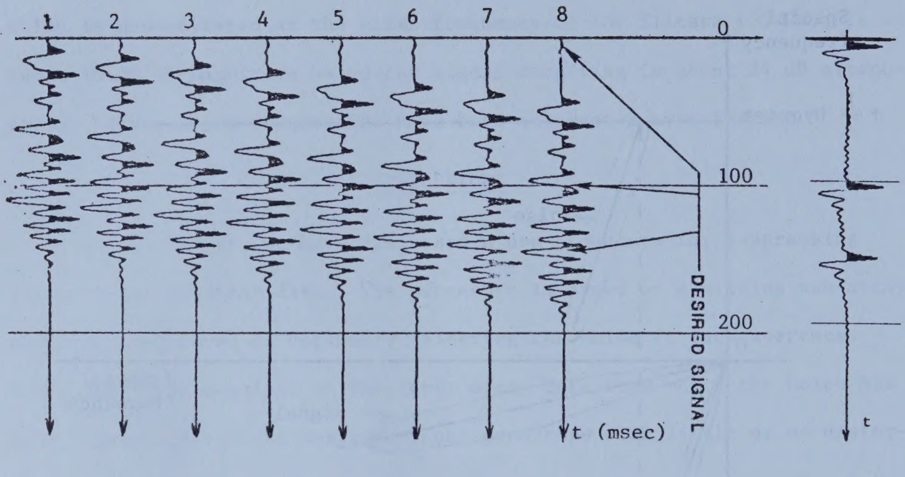

V_u
θ_u
V_d
θ_d
1
2
3
N-1
N

Figure 2: Array Signals



(a)

(b)

Figure 3: (a) Input Data
(b) Output of Optimum Filters Method

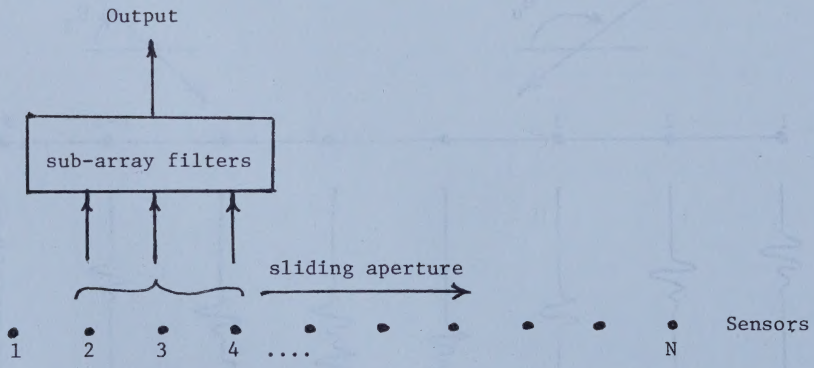

Output
sub-array filters
sliding aperture
Sensors
1
2
3
4 ....
N

Figure 4: Array Processing Scheme

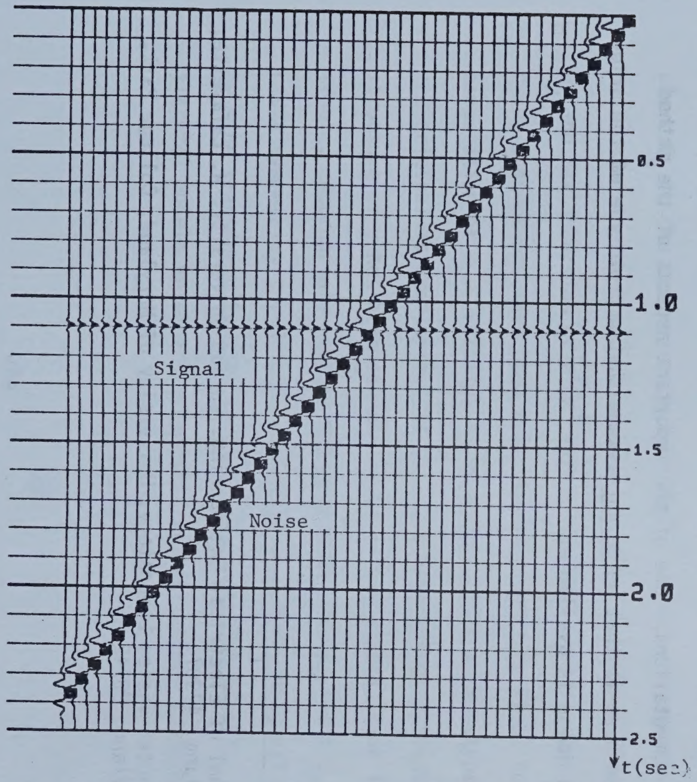


Figure 5a: Input Data

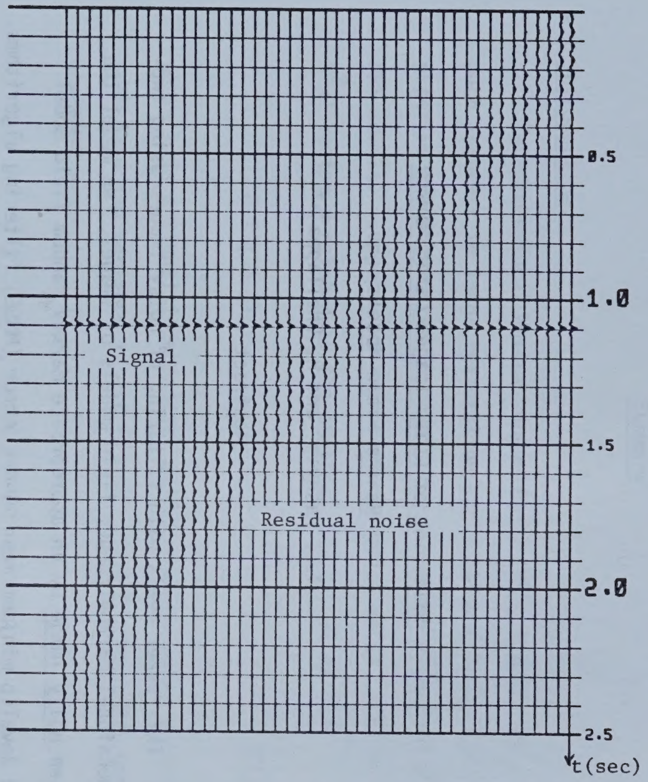


Figure 5b: Output Data

Filtering of Noisy Images Using Markov Random Field Models

R.L. Kashyap* and R. Chellappa*,+

Abstract

This paper deals with filtering of noisy images using two-dimensional Markov random field models of images. We model the given noisy image by an appropriate Markov random field model and develop minimum mean square error (MMSE) filtering algorithms. The algorithms are non recursive in structure and implemented by FFT computations. One of the important aspects of the method given here is that we do not assume that a prototype of the original image is available.

---

*School of Electrical Engineering, Purdue University, W. Lafayette, Indiana, 47907.

+Computer vision Laboratory, University of Maryland, College Park, Maryland, 20742.

Summary

a. Introduction.

Two-dimensional Markov random field (MRF) models are of much interest in image restoration application [1-3]. In this paper we are interested in developing filtering algorithm using MRF models. Suppose, we represent the given noisy image as a M^2x1 vector $\underset{\sim}{x}$ related to the original image $\underset{\sim}{y}$ as in (1):

$$\underset{\sim}{x} = \underset{\sim}{y} + \underset{\sim}{\eta}, \qquad (1)$$

where $\underset{\sim}{\eta}$ is a signal independent, additive white noise vector of mean zero and variance γ. It is well known [4, pp. 133] that the MMSE estimate $\hat{\underset{\sim}{y}}$ is given by,

$$\hat{\underset{\sim}{y}} = \underset{\sim}{Q}_y \, (\underset{\sim}{Q}_y + \gamma \underset{\sim}{I})^{-1} \, \underset{\sim}{x}, \qquad (2)$$

where $\underset{\sim}{Q}_y$ is the covariance matrix $E(\underset{\sim}{y}\underset{\sim}{y}^T)$. Two problems have to be tackled in the computation of $\hat{\underset{\sim}{y}}$, namely, the computation of $\underset{\sim}{Q}_y$ and the inversion of M^2xM^2 matrix in (2). Several simplifying assumptions have been made in the literature [1,2] for the computation of $\underset{\sim}{Q}_y$. Firstly, it is assumed that a prototype of the original is available. In [1], the underlying correlation structure of $\underset{\sim}{y}$ has been assumed to be either separable or isotropic. Appropriate nearest neighbor model for $\underset{\sim}{y}$ yields $\underset{\sim}{Q}_y$ that has a symmetric, tri-diagonal Toeplitz structure, diagonalizable by sine transforms leading to fast implementation of (2). Unilateral models which are in general approximations to underlying

MRF model have been used to develop recursive Kalman filter type algorithms in [2]. This is due to the fact that unlike in one dimensional case, the two-dimensional spectral density function (SFD) does not always factorize. Consequently, if the underlying MRF model possesses a SDF that does not factorize, any unilateral representation for this model is only approximate.

Our basic approach is as follows: Suppose that only the noise image $\underset{\sim}{x}$ is available. Then $\underset{\sim}{Q}_x$, the covariance matrix of $\underset{\sim}{x}$ is related to $\underset{\sim}{Q}_y$ as in 3:

$$\underset{\sim}{Q}_x = \underset{\sim}{Q}_y + \gamma I, \qquad (3)$$

substitution of (3) into (2) yields,

$$\hat{\underset{\sim}{y}} = (\underset{\sim}{Q}_x - \gamma I)\, \underset{\sim}{Q}_x^{-1} \qquad (4)$$

To compute $\underset{\sim}{Q}_x$, we represent $\underset{\sim}{x}$ by an appropriate MRF model characterized by parameters $(\underset{\sim}{\theta}, \nu)$ defined on a toroidal lattice [5,6]. This specific representation leads to a block circulant $\underset{\sim}{Q}_x$, whose eigenvalues are explicit functions of $(\underset{\sim}{\theta}, \nu)$. Since block-circulant matrices are diagonalized by FFT, fast implementation of (4) is possible. The FFT filtering methods given here are an improvement over the earlier methods in [4, Ch. 7] in that we do not need a prototype of the original image. In addition, the two methods are different in the characterization of signal statistic.

The estimate $\underset{\sim}{y}$ in (4) can be equivalently written in terms of the two-dimensional SDF. We use consistent estimates derived using the MRF model compared to the inconsistent estimates derived from peridogram of $\underset{\sim}{x}$ [4, Ch. 7]. We also estimate $\underset{\sim}{\gamma}$ from the given noisy image.

The specific details of the MRF model representation, computational scheme for implementing (4) and experimental results are given.

b. MRF Model on Toroidal Lattice [5,6].

Assume that $\{x(s), s\varepsilon\Omega\}$, $\Omega = \{s = (i,j), 1 \leq i, j \leq M\}$ are the the intensity levels of the noisy image $\{x(s)\}$ obeying (5-6):

$$x(s) = \sum_{r\varepsilon N_s} \theta_r \, [x(s + \gamma) + x(s - \gamma) + e(s), \; s\varepsilon\Omega_I, \qquad (5)$$

$$x(s) = \sum_{r\varepsilon N_s} \theta_r \, [x_1(s + \gamma) + x_1(s - \gamma)] + e(s), \; s\varepsilon\Omega_B, \qquad (6)$$

where

$$\begin{aligned} & x_1[s + (i,j)] \text{ with } s = (k,1) \\ & \quad = x[s + (i,j)] \text{ if } (s + (i,j)) \,\varepsilon\Omega \\ & \quad = x[(k + i - 1) \bmod M + 1, (1 = j - 1) \bmod M + 1] \\ & \qquad\quad \text{if } (s + (i,j)) \notin \Omega \end{aligned}$$

The regions Ω_B and Ω_I are defined as

$$\Omega_B = \{ s = (i,j) = s \varepsilon \Omega \text{ and } (s + r) \notin \Omega$$

for at least one $r \varepsilon N\}$, $N = \{s: s \varepsilon N_S \text{ or } -s \varepsilon N_S\}$

and

$$\Omega_I = \Omega - \Omega_B$$

The noise sequence e(s) obeys (7) [6,7]

$$E(e(s) | \text{all } y(r), r \varepsilon \Omega, r \neq s) = 0 \tag{7}$$

Hence

$$E[y(s) | \text{all } y(r), r \varepsilon \Omega, r \neq s]$$

$$= \sum_{r \varepsilon N_S} \theta_r \, (y(s + r) + y(s - r))$$

Without loss of generality we assume that

$$E(e(s)) = 0, \quad E(e^2(s)) = \nu$$

From (7), it follows that $\{e(s)\}$ is correlated as in (8):

$$\begin{aligned} E(e(s)e(r)) &= \nu, \quad s = r \\ &= -\theta_{r-s} \nu, \quad s - r \varepsilon N \\ &= 0, \quad \text{otherwise} \end{aligned} \tag{8}$$

To ensure stationarity [5],

$$\mu_s(\underset{\sim}{\theta}) = (1-2\underset{\sim}{\theta}^T\underset{\sim}{\phi}_s) > 0 \tag{9}$$

where

$$\underset{\sim}{\phi}_s = \text{Col. } [(\text{Cos } 2\frac{\pi}{M}(s - \underset{\smile}{1})^T r), r \varepsilon N_s]$$

c. MMSE Filter.

Using the vector-matrix format (5)-(6) can be written as

$$H(\underset{\sim}{\theta})\underset{\sim}{x} = \underset{\sim}{e}, \tag{10}$$

where $H(\underset{\sim}{\theta})$ is a symmetric, block-circulant matrix. From (10), the covariance matrix. $\underset{\sim}{Q}_x$ can be written as

$$\underset{\sim}{Q}_x = W(\underset{\sim}{x}\underset{\sim}{x}^T) = \nu H^{-1}(\underset{\sim}{\theta}), \tag{11}$$

Using the fact that $\underset{\sim}{Q}_x$ has eigenvalues $\nu/\mu_s(\underset{\sim}{\theta})$, $s\varepsilon\Omega$, and eigenvectors $\underset{\sim}{f}_s$, with $s = (i,j)$ defined below:

$$\underset{\sim}{f}_{ij} = \text{Col. } [t_j, \lambda_i t_j, \ldots, \lambda_i^{M-1} t_j]$$

$$\underset{\sim}{t}_j = \text{Col. } [1, \lambda_j, {}_j^2, \ldots, \lambda_j^{M-1}]$$

and

$$\lambda_i = \exp\ [\sqrt{-1}\ \frac{2\pi}{M}\ (i-1)]$$

$\hat{\underset{\sim}{y}}$ can be computed as,

$$\hat{\underset{\sim}{y}} = \frac{1}{M^2} \Sigma \underset{\sim}{f}_s \frac{(\nu - \gamma\mu_s(\underset{\sim}{\theta}))}{\nu} \underset{\sim}{f}_s^{*T} \underset{\sim}{x}, \tag{12}$$

The estimate $\underset{\sim}{y}$ is optimal if the parameters $\underset{\sim}{\theta}$, ν and γ are known exactly. In practice they are estimated from the noisy image as given below [8]:

$$\hat{\underset{\sim}{\theta}} = [\Sigma \underset{\sim}{q}(s)\underset{\sim}{q}^T(s)]^{-1} (\Sigma \underset{\sim}{q}(s)y(s)), \tag{13}$$

$$\hat{\nu} = \frac{1}{M^2} \Sigma(y(s) - \hat{\underset{\sim}{\theta}}^T \underset{\sim}{q}(s))^2, \tag{14}$$

where $\underset{\sim}{q}(s) = \text{Col.}\ [y(s + r) + y(s - r),\ r\epsilon N_S]$.

An estimate of γ for 64 x 64 image is obtained as the steady state component the SDF of $\underset{\sim}{x}$ [4, pp. 144] as in (15):

$$\hat{\gamma} = \frac{1}{16} \sum_{s\epsilon\Omega_u} S_x(s), \tag{15}$$

where $\Omega_u = \{s : s = (i,j),\ \ 29 \leq i,\ \ j \leq 32\}$

and $\quad S_x(s) = \nu/\mu_s(\hat{\underset{\sim}{\theta}}),\ s\epsilon\Omega.$

The estimates $\underset{\sim}{\theta}$, $\hat{\nu}$ and $\hat{\gamma}$ are used in (11), in place of $\underset{\sim}{\theta}$, ν and γ.

d. Experimental Results: A 64 x 64 window of the original girl image in Fig. 1 was corrupted by an additive noise of signal to noise ratio (SNR) = 5, where SNR is defined as the ratio of signal to noise variance. Several MRF models were fitted to the noisy image and the corresponding filtered images are in Fig. 2. The details of models fitted are given in the figure caption. The improvement is noticeable in the filtered images corresponding to models N_{S3} and N_{S4}. The MSE between $\underset{\sim}{y}$ and $\underset{\sim}{\hat{y}}$ obtained from the best filter N_{S4} is 91.064 corresponding to the MSE of 378.53 between $\underset{\sim}{y}$ and $\underset{\sim}{x}$. The best estimate of γ obtained was 378.74 using the model N_{S4} compared to the true value 393.01.

e. *Conclusions*. We have given an MMSE filtering scheme using MRF models. We have not made any simplifying assumptions like separable correlation function or isotropy. We use more accurate estimates of SDF than those derived from periodogram. More significantly, we do not assume that a prototype of the original is available. We have assumed a toroidal lattice representation for the image $\underset{\sim}{x}$. It is a natural question to ask if this assumption is reasonable. We argue in the affirmative as given a toroidal lattice representation an infinite lattice representation can be constructed by letting M tend to infinity [9]. Secondly, the numerical values of the autocorrelation function evaluated at lower lags corresponding to a toroidal lattice and the infinite lattice are close to each other indicating that the error in approximation due to toroidal lattice representation is negligible.

1. A. K. Jain, "A fast Karhumen-Loeve transform for digital restoration of images degraded by white and colored noise", IEEE Trans. Computers, Vol. C-26, pp. 560-571, June 1977.

2. J. W. Woods, "Markov image modeling", IEEE Trans. Automat. Control, Vol. AC-23, pp. 846-850, Oct. 1978.

3. R. K. Kashyap and R. Chellappa, "Image restoration using random field models of images", Proc. of Eighteenth Allterton Conf. On Commn. Control and Computing, Univ. of Illinois, Urbana, pp. 956-965, Oct. 1980.

4. H. C. Andrews and B. R. Hunt, Digital Image Restoration, Prentice Hall Inc., N.J., 1977.

5. R. L. Kashyap, "Random field models on toroidal lattices for finite images", Proc. 5th Int. Conf. on Pattern Recognition, Miami, Florida, pp. 1103-1105, Dec. 1980.

6. R. L. Kashyap, "Random field models on finite lattices for finite images", Presented at the Conference on Information Sciences, John Hopkins University, 1981.

7. J. W. Woods, "Two-dimensional discrete Markov random fields", IEEE Trans. Inform. Theory, Vol. IT-18, pp. 232-240, Mar. 1972.

8. R. Chellappa, "Fitting Markov random field models to images", TR-994, Computer Sciences Ctr., Univ. of Maryland, College Pk., Md., 20742, Jan. 1981.

9. P.A.P. Moran, "A Gaussian Markovian process on a square lattice", J. Appl. Prob., Vol. 10, pp. 605-612.

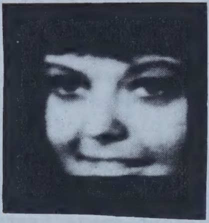


Fig. 1. Original Image

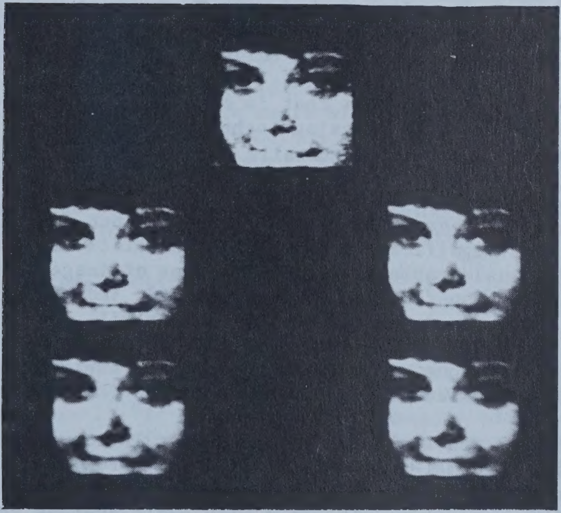


Fig. 2. Results of Filtering. Top: Noisy image of SNR = 5. Second row: left $N_{S1} = \{(0,1), (1,0)\}$, right $N_{S2} = \{(0,1), (1,1), (1,0)\}$. Third row: left $N_{S3} = \{(0,1), (1,0), (1,1), (1,-1)\}$, right: $N_{S4} = \{(0,1), (1,0), (1,1), (1,-1), (0,2), (2,0)\}$.

AN EFFICIENT SOURCE CODING TECHNIQUE FOR DATA COMPRESSION OF MULTILEVEL DIGITIZED IMAGES

C.B. FONG AND M. KANEFSKY
Department of Electrical Engineering, University of Pittsburgh
Pittsburgh, PA 15261

Abstract. An efficient source coding technique that is useful for the compression of multilevel digitized images is presented. Differential Bit Plane Coding involves compressing each bit plane independently by DPCM coding based on maximum likelihood prediction. DBPC has been found to be very efficient for information preserving coding of multilevel digitized images. DBPC is extended here to an efficient information reducing coding technique, which preserves the important image features.

1. INTRODUCTION

The rapid growth in the desire to store images of all kinds is responsible for the need to compress digitized images. Compression algorithms are usually divided into two categories, information perserving (IP) and information reducing (IR). IP techniques have the desirable property that the original image can be exactly reconstructed. Unfortunately these techniques rarely result in very much compression with bit rate reductions of more than 40% being rarely accomplished. Considerably more compression can be achieved if one uses an IR coding technique. The reconstructed image after IR compression is "nearly the same" as the original image. Unfortunately the question of whether an image is "close enough" to the original is often highly subjective.

One possible measure of closeness that is often used is the mean square error. Indeed, many of the IR compression techniques seek optimality in the sense of minimum m.s.e. which is a totally inadequate measure of closeness for almost all images. It is particularly significant only for the class of stationary gaussian images which is virtually nonexistant. Not only are most image histograms very ungaussian [17] but images are rarely stationary. Most significantly an image with relatively small mean square error can have considerable smoothing at edges of image objects which are often the most significant features. Indeed, techniques that are near optimal in the sense of minimum mean square error will tend to smooth edges. While other measures such as the average error magnitude or peak error may be more relevant, one often resorts to purely subjective comparisons such as displaying the reconsturcted images of two IR techniques side by side.

This paper discusses and evaluates and efficient IR compression technique that employs differential-pulse-code-modulation (DPCM) and a block truncation algorithm [21]. This technique is partially information preserving and as a consequence it results in a very small average error magnitude and tends to preserve important features such as object edges.

DPCM has been proposed [1-8] as an efficient IP image compression technique. Because of implementation considerations, this procedure usually compresses each of the binary bit planes [11,12] independently of the others, called Differential Bit Plane Coding (DBPC) [20]. First a Markov mesh model [9] is identified [10] for each plane which permits a simple maximum likelihood prediction of the binary image pixels which minimizes the probability of error. When the predicted value differs from the actual binary value, a one is stored. This difference which is mainly zero's is encoded by a very efficient IP technique called Gordon-Block (GB) encoding [19]. The most significant planes are highly compressed but the least significant bit planes have entropies of nearly 1 bit per pixel and are only slightly compressible. The structure of DBPC will be discussed in more detail in section 2.

This structure of treating the bit planes seperately makes it easy to modify DPCM to an efficient IR technique. If IP coding is used for the most significant planes and a low bit-rate information reducing algorithm for the least significant planes, small peak errors are guaranteed and image features are preserved. For the least significant bit planes we employ a block truncation algorithm which preserves some statistical properties of these planes. This IR technique based on DBPC will be discussed in section 3.

The IR technique is compared with a standard transform IR compression technique [18] for real images. For the same degree of compressibilty, DPCM with block truncation on the least significant planes has a smaller mean square error, a smaller average error magnitude, and subjectively looks much better with object edges clearly preserved.

2. DIFFERENTIAL BIT PLANE CODING

Differential Bit Plane Coding involves the following steps as shown in Figure 1. First, the multilevel image is split into its bit planes. The individual bit planes are modeled as binary Markov mesh's with strong spatial dependency. Second, each bit plane model is identified which permits a simple maximum likelihood prediction of the binary image pixels which minimizes the probability of error. The image pixel value is predicted according to its state from an identified transition table. Finally, the difference between the prediction and the actual value is stored and encoded by a very efficient IP technique called Gordon Block encoding [19]. The strategy of GB encoding is to decompose the binary image into smaller blocks with some reasonable small block size for a pratical implementation and to encode each block by Gordon's algorithm [13-16]

Assume the nxn "difference" image has L levels. For GB encoding with block size S of the i-th bit plane with m_j, the number of ones in the j-th block, m_q the number of ones in the remainder block with size R, and u, the number of blocks, one needs b_i bits where

$$b_i = u\log_2 S+\log_2 R + \sum_{j=1}^{u}\sum_{k=1}^{m_j}\log_2\left(\frac{S+1-k}{m_j+1-k}\right)+\sum_{k=1}^{m_q}\log_2\left(\frac{R+1-k}{m_q+1-k}\right). \qquad (1)$$

In order to employ the same maximum likelihood prediction algorithm in the coder and decoder, the prediction algorithm has tobe encoded and stored. To encode these maximum likelihood values corresponding to each state requires $2^{Q(K_i)}$ bits, called bookkeeping bits, where $Q(K_i)$ is the number of elements in the state of the K-th order i-th bit plane. Neglecting the fact that the border estimates are nonoptimal, the total bits (including the bookkeeping bits) B to encode a nxn, L level image is given by

$$B = \sum_{i=1}^{\log_2 L} 2^{Q(K_i)} + \sum_{i=1}^{\log_2 L} b_i . \qquad (2)$$

We define the compressibility as the total number of bits required to store an image divided by the number of bits in an uncompressed image. Using this IP technique, the compressibility, C is given by

$$C = \left(\sum_{i=1}^{\log_2 L} 2^{Q(K_i)}\right)/n^2 \cdot \log_2 L + \left(\sum_{i=1}^{\log_2 L} b_i\right)/n^2 \cdot \log_2 L . \qquad (3)$$

The first term is the effect of bookkeeping bits. For example, a "chest"

X-ray and "hand" X-ray (each having 256 levels) are shown in Fig. 2 and 3 respectively. Using the DBPC technique, the compressibility of these two images are 0.587 and 0.526, which corresponds to a saving of 41.33% and 47.37%. In order to achieve greater compression, we seek to modify this algorithm to one that is information reducing.

3. AN IR TECHNIQUE BASED ON DBPC

For an image digitized with a conventional binary code, errors which occur in the most significant bit planes are more serious than those in the least significant bit planes. If IP coding (DBPC) is used for the most significant bit planes and a low bit rate information reducing algorithm for the least significant planes, small peak error are guaranteed and image features are preserved. Furthermore, since the most significant bit planes are highly compressable, the cost to preserve these significant planes is low. For the least significant planes, we employ a block truncation algorithm [21] which preserves some statistical properties of these planes. If the least significant planes are simply discarded, this IP technique for the most significant planes can also be regarded as a form of logarithmic quantization. When the dynamic range varies widely, one can employ block truncation on some planes and discard the rest.

The block truncation algorithm for the least significant planes involves the following steps. First the partial image containing the least significant planes is divided into nonoverlapping blocks of size kxk. Second, the sample mean and the sample variance of each block are determined and quantized. Third, the pixels of each block are classified into two classes by comparing with the threshold (sample mean), which yields a "threshold" binary image (or bit plane). Finally, the quantized sample means and sample variances of all blocks are split into their bit planes and along with the "threshold" bit plane are encoded by the GB encoding technique and stored. In the reconstruction of each block, the pixels above the threshold (indicated by "threshold" bit plane) are set to be

$$b = \bar{X} + \bar{\sigma}\sqrt{\frac{k^2-q}{q}}, \qquad (4)$$

and those below the threshold are set to be

$$a = \bar{X} - \bar{\sigma}\sqrt{\frac{q}{k^2-q}} \qquad (5)$$

where $\bar{X}$ is the sample mean, $\bar{\sigma}$ is the sample variance, and q is the number of pixels in each kxk block that are above the threshold. Note that the sample mean and sample variance are preserved in the reconstructed partial image.

4. AN EXAMPLE AND COMPARISON

A digitized 256 level image of Pittsburgh's "Golden Triangle" is shown in Fig. 4. We use different methods to compress this image, which are listed below.

A. DBPC for the two most significant planes with block truncation of the other 6 planes. These two most significant planes are identified as first order Markov meshs and their difference bit planes from maximum likelihood prediction are endoded by GB encoding with block size 64. Note that GB encoding with block size 64 in DBPC is also employed in method C and D. The block size 4x4 is used for the block truncation of the other 6 planes. The sample means and sample variances of these blocks are quantized by a 4 bit quantizer. These quantized sample means and variances are split into their 4 bit planes and along with the "threshold" bit plane are encoded by GB encoding with block size 64.

B. Transform encoding.

C. DBPC for the two most significant planes with the others discarded.

D. DBPC for the four most significant bit planeswhich are identified as first order Markov meshs. The other planes are discarded.

E. Block truncation coding on all planes. The block size 4x4 is used for the block truncation algorithm. A 10 bit quantizer is chosen for quantizing the sample means and variances in order to achieve the bit rate which is close to that of method A. The quantized sample means and variances, along with the "threshold" bit plane are stored uncompressed.

The resulting bit rate, mean square error, and mean absolute error of all methods are listed in Table 1. The reconstructed images are shown in figures 5,6,7,8 and 9 for method A, B, C, D and E. In method A, C and D, the compressibility of the two most significant planes are 0.382 and 0.503 which indicates that they are highly compressible. We found that the edges in the reconstructed images of method C and D are clearly preserved. For comparing method A and B (transform encoding), we use the same bit rate, 2.31 bits/pixel which corresponds to a saving of 71.13%. We see that DPCM with block truncation of the least significant planes has a smaller mean square error, a smaller mean absolute error, and subjectively looks much better with object edges clearly preserved.

REFERENCES

[1] Cutler, "Differential Quantization of communication Signals", U.S. patent 2 605 36., July 1952.

[2] Graham, "Prediction Quantizing of Television Signals", IRE WESCON Conv. Rec. 1958.

[3] O'Neal, "Predictive Quantizing Systems (Differential Pulse Code Modulation) for the Transmission of Television Signals", Bell System Tech. Journel, March 1971.

[4] Connner, Pease and Scholes, "Television Coding Using Two-Dimension Spatial Prediction", Bell System Tech. Journel, March 1971.

[5] Rice and Plaunt, "Adaptive Variable-Length Coding for Efficient compression of Space Craft Television Data", IEEE Tranon Comm. Tech. Dec., 1971.

[6] Millard and Maunsell, "Digital Encoding of the Video Signal", Bell System Tech. Journal, 1971.

[7] Habibi, "Comparison of Nth-Order DPCM Encoder with Linear Transformation and Block Quantization Techniques" IEEE Tran. on Comm. Tech. Vol. Com-19, No. 6, Dec., 1971.

[8] Zschunke, "DPCM picture Coding with Adaptive Prediction", IEEE, Tran. on Comm., Vol. Com-25, No. 11, 1977.

[9] Abend, Harley and Kanal, "Classification of Binary Rsndom Patterns", IEEE Trans. on Information Theory, Oct., 1965.

[10] Sahasrabudhe, "Identification of a Markov Mesh Model and Its Application to discrete Image Processing", Ph.D. Dissertation-University of Pgh., 1979.

[11] Schwartz and Barker, "Bit-Plane Encoding: A Technique for Source Encoding", IEEE Tran. on Aerospace and Electronic Systems, 1966.

[12] Kanefsky, "On the Compression and Recognition of Two-Dimension Patterns from their Bit Planes", Proc. of the 2nd Hawaii Int., Conf. on System Sciences 1969.

[13] Gordon, "An Algorithm Code for Montonic Sources", Final Report of NASA GSFC Summer Workshop, 1965.

[14] Lynch, "Data Compression with Error Control Coding for Space Telemetry", Ph.D. Thesis - Univ. of Maryland, 1966.

[15] Lynch, "SequenceTime Coding for Data Compression", Letter to Proc. IEEE, Oct., 1966.

[16] Davisson, "Comments on Sequence Time Coding for Data Compression", Letter to Prc. IEEE, Dec 1966.

[17] Netravali and Limb, "Picture Coding", Proc. of IEEE, Vol. 68, No. 3, Mar., 1980.

[18] Gonalez and Wintz, "Digital Image Processing", 1977.

[19] Fong and Kanefsky, "An Efficient Source Coding Technique for Random Binary Images", Proceedings of Conference on Information Science and Systems, Baltimore, Maryland, March 1981.

[20] Kanefsky and Fong, "DPCM Based on Maximum Likelihood Prediction for Data Compression of Digitized Images", to be published.

[21] Delp and Mitchell, "Image Compression on Using Blocking Truncation Coding", IEEE Tran. on Comm. Vol. COM-27, No. 9, Sept. 19791

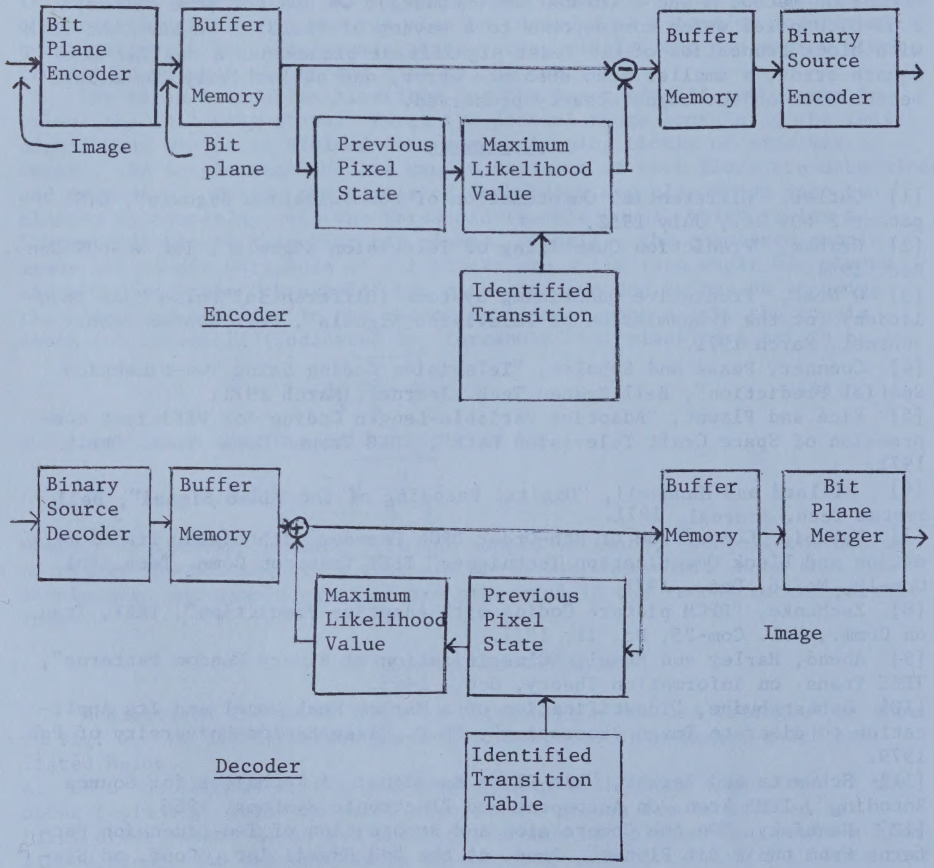

Bit Plane Encoder
Buffer Memory
Buffer Memory
Binary Source Encoder
Image
Bit plane
Previous Pixel State
Maximum Likelihood Value
Identified Transition
Encoder
Binary Source Decoder
Buffer Memory
Buffer Memory
Bit Plane Merger
Maximum Likelihood Value
Previous Pixel State
Image
Decoder
Identified Transition Table

Figure 1: The Diagram of the DBPC

Method	bit rate (bits/pixel)	M.S.E.	M.A.E.
A	2.31	102.32	6.96
B	2.31	204.36	9.64
C	0.90	1301.92	30.27
D	2.35	75.01	7.20
E	2.25	379.23	11.12

Table 1: Resuts of Different Methods

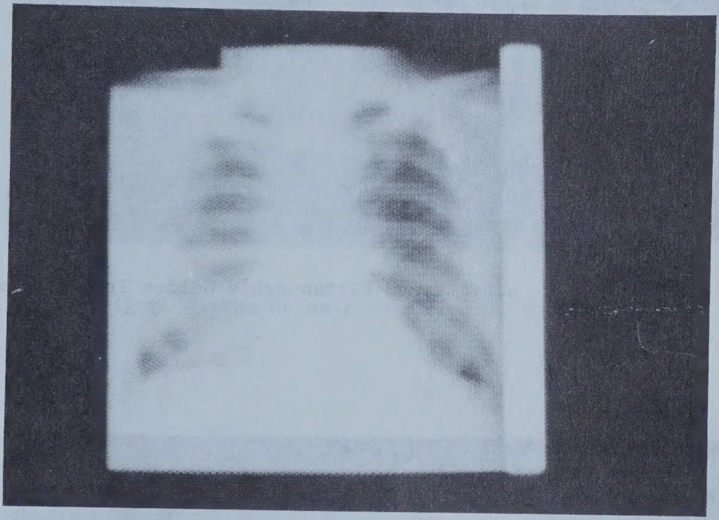


Figure 2: A Digitized Chest X-ray

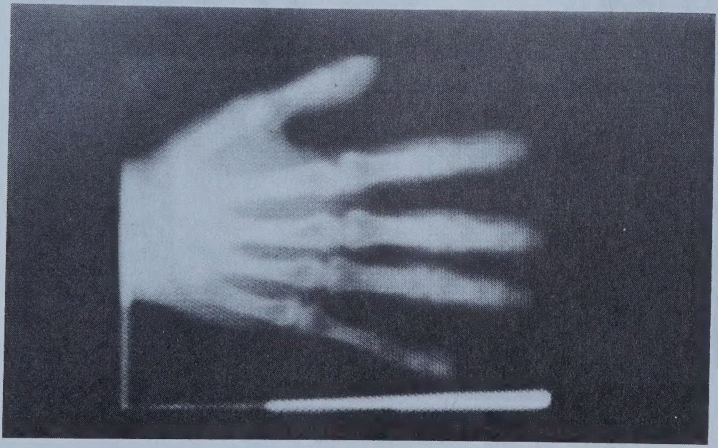


Figure 3: A Digitized Hand X-ray

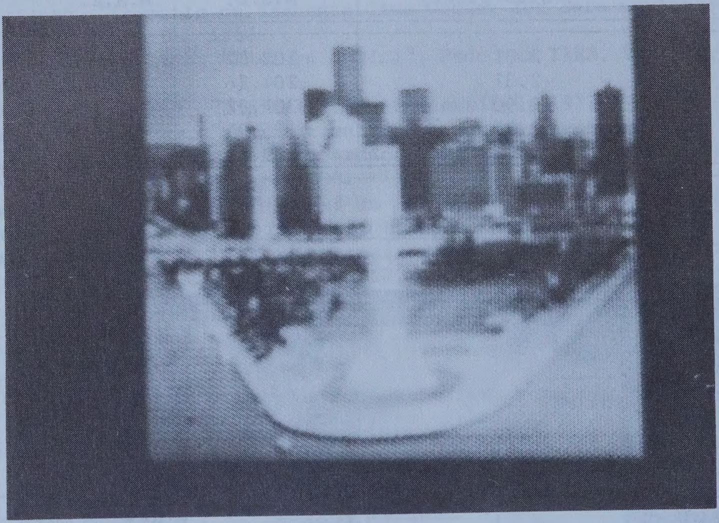


Figure 4: A Digitized Pittsburgh's Golden Triangle Image

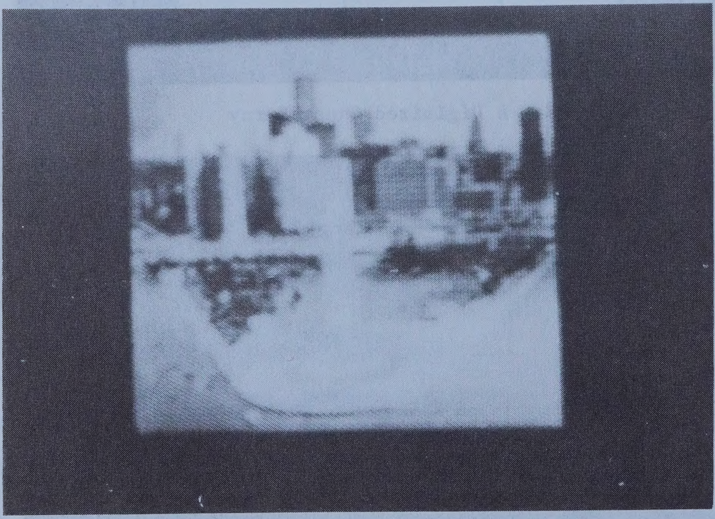


Figure 5: The Reconstructed Image from Method A

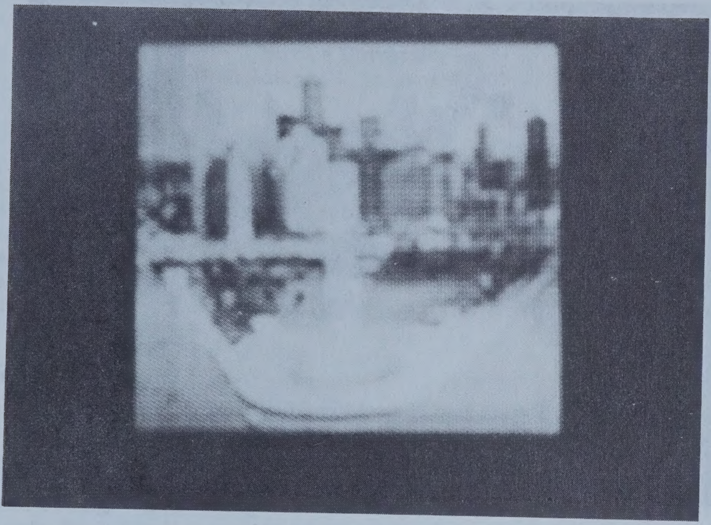


Figure 6: The Reconstructed Image from Method B

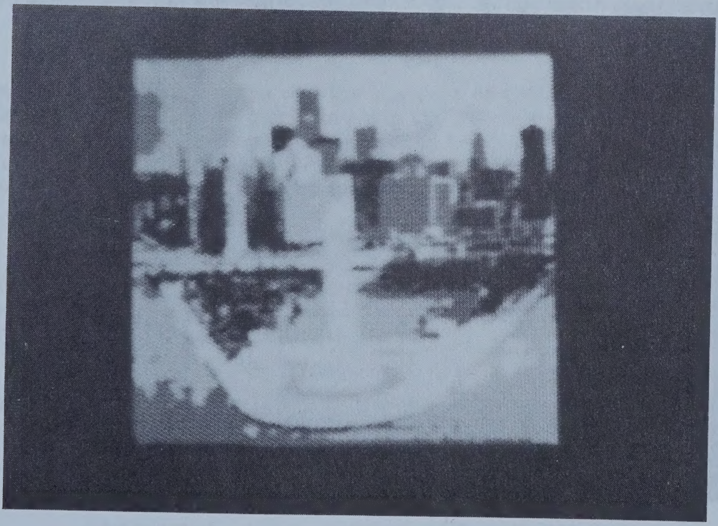


Figure 7: The Reconstruction Image from Method C

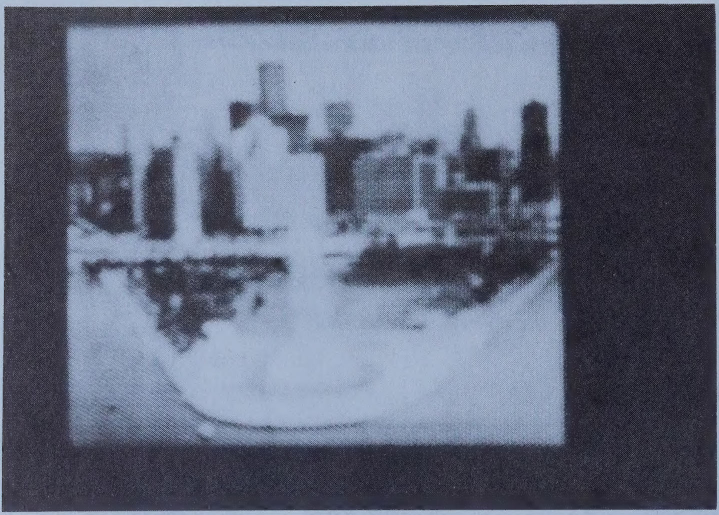


Figure 8: The Reconstructed Image from Method D

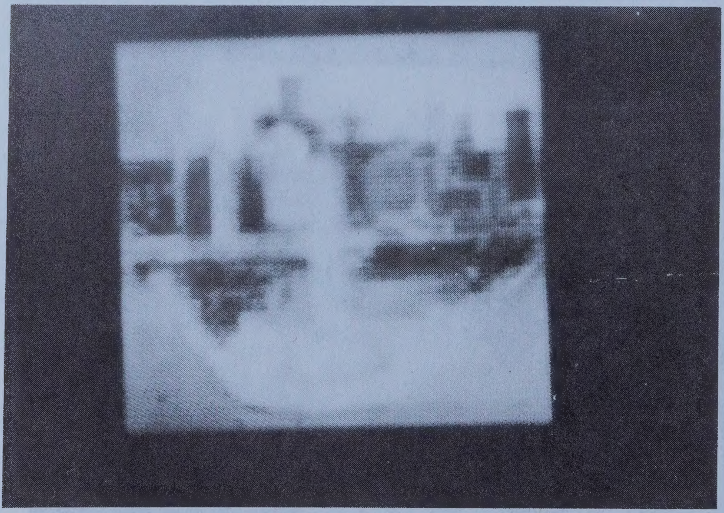


Figure 9: The Reconstructed Image from Method E

IMAGE SOURCE CODING USING MEDIAN FILTER ROOTS

G. R. ARCE and N.C. GALLAGHER, jr
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

ABSTRACT

In this paper we source encode the truncated block used in *Block Truncation Coding*. It is shown that the truncated block is well aproximated by wide sense markoff statistics; a signal having this characteristics has a high probability of belonging to the *Root Signal Set* of median filters. Since the *Root Signal Space* is smaller than the binary space, it takes fewer bits to specify the truncated block in the *Root Signal Space*, obtaining in this manner rate compression.

1. INTRODUCTION

Block Truncation Coding is a coding method that has been proposed by Mitchell & Delp for coding images in the spatial domain [1]. Its performance and simplicity make this coding method very attractive for real time image transmission. The bit rate attainable with standart *BTC* is about 1.63 bit/pel for intraframe coding. The *BTC* algorithm is simple, the image is divided into blocks of 4 × 4 pels. Each block is quantized into 2 levels (1 bit/pel), the threshold level is chosen to preserve the first 2 moments of the block. The block is transmitted as a binary plane along with information on the mean and the standard deviation of the block. At the receiver, the samples which have a value above the threshold are set to

$$b=\eta+\sigma\sqrt{\frac{k-q}{q}}, \tag{1}$$

and the samples lying below the threshold are set to

$$a=\eta-\sigma\sqrt{\frac{q}{k-q}}, \tag{2}$$

where η and σ denote the mean and the standart deviation of the block respectively, k is 16 (total # of samples), and q is the number of samples lying above the threshold level. The process is illustrated in fig. 1 (a),(b),(c), where the original block, the bit plane, and the reconstructed block are shown respectively. The truncation process is just a hard limiter as shown in fig. 2. In general, image signals are modeled as having exponential correlations of the form

$$R(\Delta x,\Delta y)=e^{-\alpha|\Delta x|-\beta|\Delta y|},$$

where Δx and Δy are the distances between the pels, in the x and y direction. Since here we only discuss one dimensional filtering, the input signal to the hard limiter has an autocorrelation of the the form

$$R(\Delta x)=e^{-\alpha|\Delta x|}, \tag{3}$$

where α ranges between 0.05 to 0.35 , depending on the image.

10	10	11	12		1	1	1	1		12	12	12	12
10	9	12	15		1	0	1	1		12	3	12	12
3	4	12	15		0	0	1	1		3	3	12	12
2	4	10	15		0	0	1	1		3	3	12	12

a) Original Block b) Truncated Block c) Reconstructed

Fig. 1 Truncating Process : η =9.6 ,σ =4.2 , q= 11 ,a=3 ,and b=12.

---

[*] The authors greatefully acknowledge the support of National Science Foundation under Grant # ECS 8024302

If we let the input image be normal with correlation as shown in (3),the correlation of the output plane can be shown to be [2]

$$R_{yy}(\Delta x)=\frac{2}{\pi}\sin^{-1}(e^{-\alpha|\Delta x|}). \tag{4}$$

We see that the bit plane retains some correlation. Since each binary block has to be sent separately, we cannot use run length coding [3] or any other standard scheme to encode the bit plane, instead we use median filters.

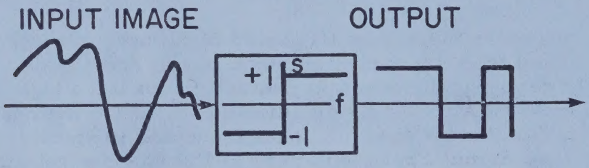
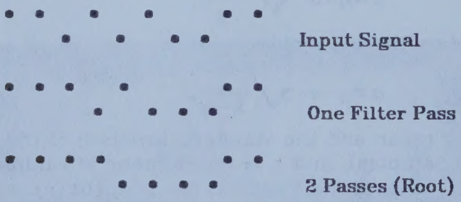


Fig. 2 Hard Limiting Model.

Median filtering is a very simple nonlinear digital operation. Briefly it computes the median of 3 adjacent samples spanned by the filters window and replaces the center sample by the calculated median. This filter has the property that if a signal is repeatedly median filtered, the output after each filter pass converges to a signal invariant to further median filtering. The signal set invariant to the filter is called "*The Root Signal Set.*" A description of this signal set can be found in [4]; for our purposes it is enough to mention that this set is much smaller than the binary space. The next example will illustrate the one dimensional median filtering process.

Example 1



2. ROOT CODING

The idea of using *Median Filters* to encode the bit plane is to median filter the truncated block into a root signal and instead of transmiting the block in the binary domain, send a code specifying the root. Signals in the bit plane are just hard limited normal signals; as such they are approximated as wide sense markovian. Thus, it is shown that the block signals have a very high probability of belonging to the *Root Signal Set.* Since the *Root Signal Set* is much smaller than the general binary space, we need fewer bits to represent a block in the *Root* domain; hence, we obtain compression. In the process of median filtering the bit plane, we have the option of filtering the block in two dimensions (obtain a single root for the entire block), or filter in one dimension (obtain one root for each row of the block).The next example shows the one dimensional filtering process for the block shown in fig.1.

Example 2

BTC Block	Filtered Block	Reconstructed
1 1 1 1	1 1 1 1	12 12 12 12
1 0 1 1	1 1 1 1	12 12 12 12
0 0 1 1	0 0 1 1	3 3 12 12
0 0 1 1	0 0 1 1	3 3 12 12

The tree structure of the 2 dimensional root signal set is very complicated; here we will present 2 dimensional results for illustration purposes only. First, let us describe the one dimensional process. Each row of the block consists of four bits, with conventional *BTC* we need 16 bits to specify the bit plane. Using our *Root Space* along with the statistical nature of the bit plane, we only require 12 bits, reducing in this way the bit rate from 1.63 to 1.31 bits/pel. Using two dimensional filtering, we require about 13 bits to describe the block. Figure 3 shows the PCM figure, the two dimensional *Root* coded picture, the conventional *BTC* ,and the 1 dimensional *Root* coded picture (top: left,right; bottom: left, right respectively) † .

3. MATHEMATICAL MODEL FOR THE TRUNCATION PROCESS

In order to evaluate this coding method, we must look analytically and experimentally at the distortion introduced by this process. The first part of the analysis is the modeling of the one dimensional filtering of the image. The occurrence probability distribuition for each of the signals is closely related to the distribution of the duration of excursions of normal fluctuations. *Rice* [5] derives an aproximate probability density of having an upper excursion of length τ , given that the excursion started at the origin. For such distribuition , the expected number of zero crossings in a finite interval is proportional to

$$\frac{1}{\pi}\left[-\frac{\psi''(0)}{\psi(0)}\right]^{1/2}$$

Where $\psi(\tau) = R(\tau)$ is the autocorrelation function . Since

$$\psi_0'' = \frac{\partial^2 e^{-\alpha|\tau|}}{\partial \tau^2} \to \infty \quad as \quad \tau \to 0,$$

the expected number of crossings is not defined. This shows that we cannot treat the input image as a continuos signal,but as a discrete random process. Denote the input image samples by $\{ f_n ,f_{n+1} ,f_{n+2} , f_{n+3} \}$ and their truncated binary versions by $\{ s_n , s_{n+1} , s_{n+2} , s_{n+3} \}$ (row of the truncated block). Let the 4 image samples, be jointly normal. Obtaining the ocurrence probability for the root signal set amounts to integrating the 4-variate normal distribuition within the limits specified by the root sequence. A simpler model can be obtained by observing that the binary plane can be described aproximately by a first order markoff process [6]. For the markoff model,

$$\mathbf{Pr}(s_{n+3}=0, s_{n+2}=0, s_{n+1}=1, s_n=1)= \tag{5}$$

$$\mathbf{Pr}(s_{n+3}=0/s_{n+2}=0)\mathbf{Pr}(s_{n+2}=0/s_{n+2}=1)\mathbf{Pr}(s_{n+1}=1/s_n=1)\mathbf{Pr}(s_n=1).$$

Where again $\{ s_n ,s_{n+1} ,s_{n+2} , s_{n+3} \}$ are the samples in a row of the truncated block. In order to obtain the probability of occurrence of any signal in the plane, we only need the transition probablities . Denote the probability of a change of bit value between two consecutive pels as

$$P_1=\mathbf{Pr}(s_{n+1}=0/s_n=1)=\mathbf{Pr}(s_{n+1}=1/s_n=0),$$

and

$$P_0=1-P_1$$

the probability of a sequence of two equal valued bits. Following the markoff model, and letting the random variables f_n and f_{n+1} be jointly normal with correlation coefficient ρ , then P1 is the area spanned by the bi-variate normal density in the second and fourth quadrant. In this case

$$\rho = e^{-\alpha}; \quad \beta = \cos^{-1}\rho ,$$

then it is simple to show that

$$P_1 = \frac{\beta}{\pi} , \text{ and } P_0 = 1-\frac{\beta}{\pi}. \tag{6}$$

---

† The original in this coding analysis is the conventional BTC image.

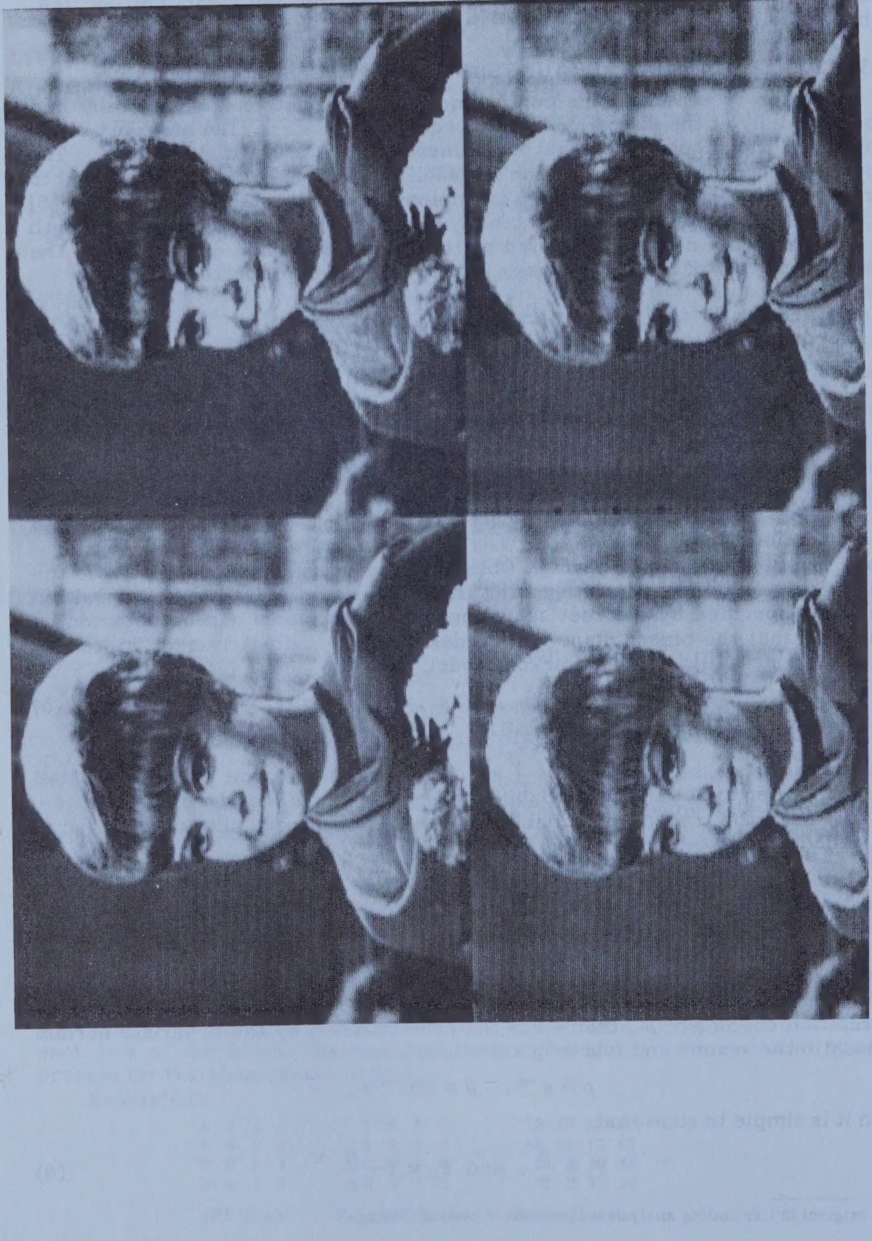


Fig. 3 PCM, 2 Dim. Root Coded (top: L,R) ; BTC, 1 Dim Root Coded (bottom: L,R)

Table 1 shows each root sequence along with its probability of ocurrence. As an illustration for $\alpha = .15$, **Pr** (Row $\in$ Root) = 0.81 and **Pr** (Row $\notin$ Root) =0.19 . The analytical results are in agreement with the experimental probabilities that we have computed. Table 1 also shows the theoretical values and the experimental values for each probability of occurrence.

TABLE 1 Probability Distribuition for the Root Space.

Root #	Sequence	Pr(root/a)	Pr(root/a=0.2)	Pr(root/Fig 5)
1	0000	$1/2\ P_0^3$	0.256	0.20
2	0001	$1/2\ P_0^2\ P_1$	0.064	0.08
3	0011	$1/2\ P_0^2\ P_1$	0.064	0.11
4	0110	$1/2\ P_0\ P_1^2$	0.015	0.02
5	0111	$1/2\ P_0^2\ P_1$	0.064	0.07
6	1000	$1/2\ P_0^2\ P_1$	0.064	0.09
7	1001	$1/2\ P_0\ P_0^2$	0.015	0.02
8	1100	$1/2\ P_0^2\ P_1$	0.064	0.12
9	1110	$1/2\ P_0^2\ P_1$	0.064	0.09
10	1111	$1/2\ P_0^3$	0.256	0.20

4. DISTORTION ANALYSIS

By selecting different subsets within the root signal set, we can obtain higher data compresion at a cost of higher probability of distortion. In order to evaluate the different subsets, we will locate them in a rate distortion plane along with an optimum bound function. To calculate the rate distortion function, we require the knowledge of the bit plane autocorrelation; a derivation of it is in order.

As before, we model the bit plane as a markoff process. For mathematical simplicity define a random variable **x** such that

$$\mathbf{x_n} = \begin{cases} 1 & \text{if } s_n = 1 \\ -1 & \text{if } s_n = 0 \end{cases}$$

Next, define a two state markoff space, where the state is the value of **x**

$$state\ 0 : x_n = 1$$
$$state\ 1 : x_n = -1$$

The markoff transition matrix is

$$\mathbf{P} = \begin{bmatrix} P_0 & P_1 \\ P_1 & P_0 \end{bmatrix},$$

where Po and P 1 are given in (6). The k step transition probability matrix is given by

$$\mathbf{P}^k = \frac{1}{2} \begin{bmatrix} \lambda_1^k + \lambda_2^k & \lambda_1^k - \lambda_2^k \\ \lambda_1^k - \lambda_2^k & \lambda_1^k + \lambda_2^k \end{bmatrix},$$

where

$$\lambda_1 = P_0 + P_1 \quad \text{and} \quad \lambda_2 = P_0 - P_1$$

are the eigenvalues of **P** . Denote the probability of a bit having a value of zero in the truncated block by Qo, and the probability of it being one by Q1. The autocorrelation of the truncated block is

$$\mathbf{E}(\mathbf{x_i x_j}) = (Q_0 P_{00}^{j-i} + Q_1 P_{11}^{j-i}) - (Q_0 P_{01}^{j-i} + Q_1 P_{10}^{j-i}). \qquad (7)$$

Since the threshold of the truncation is the mean of the block and the input distribuition about the mean is symetric, Qo = ½ . The autocorrelation is then simplified to

$$R(i,j) = \lambda_2^{|j-i|} = (P_0 - P_1)^{|j-i|}. \qquad (8)$$

For sources with memory, the rate distortion function *R(D)* is known exactly for only a few special cases. On the other hand, bounds to *R(D)* are simple to calculate for stationary and ergodic sources.

Define the square error distortion measure as

$$d\,(x_i - x_f) = \sum (x_i - x_f)^2 \ , \textit{where } x_i \text{ and } x_f \qquad (9)$$

are the input and filtered samples of the block respectively. Then, for any discrete zero mean source with spectral density Φ and squared error distortion measure [7],

$$R(D) \leq \frac{1}{4\pi}\int_{-\pi}^{\pi} max[0, ln\frac{\Phi(\omega)}{\Theta}]\,d\omega \ , \qquad (10)$$

where Θ satisfies

$$D = \frac{1}{2\pi}\int_{-\pi}^{\pi} min[\Theta, \Phi(\omega)]\,d\omega \ \text{ and } \ \Phi(\omega) = \sum_{k=-\infty}^{\infty} \varphi_k\, e^{-ik\omega} \ , with \ \ \varphi_k = E(x_i, x_{i+k}).$$

In this case,

$$\varphi_k = \lambda_2^k \ , \text{ and } \Phi(\omega) = \sum_{k=-\infty}^{\infty} \lambda_2^k\, e^{-ik\omega}.$$

Simplifying Φ we get

$$\Phi(\omega) = \left[\frac{1 - \lambda_2^2}{1 - 2\,\lambda_2 \cos\omega + \lambda_2^2}\right]. \qquad (11)$$

For

$$\Phi < \frac{1-\lambda_2}{1+\lambda_2} \ , \ \Phi = D.$$

Placing (11) into (10), the rate distortion bound is obtained [7]

$$R(D) \leq \frac{1}{2}\log_2 \frac{(1-\lambda_2^2)}{D} \quad \text{for } 0 \leq D < \left[\frac{1-\lambda_2}{1+\lambda_2}\right].$$

The distortion introduced by the filtering process is

$$D_f = \sum_{seq_j}\left[\sum_{x_i \varepsilon\, seq_j} (x_i - x_f)^2\right] P(seq_j), \qquad (12)$$

where the inner term is the distortion measure for the j th sequence and the last term denotes the probability of occurrence for such j th sequence . Recall that x takes on the values +1 and -1 , then for each bit that is changed by the filtering process, the distortion is 4. Since the sequences that belong to the root signal space introduce no distortion ,and by noticing that all the non root signals differ from their filtered version by only one bit , (12) can be written as

D= 4 P(Row ∉ root) . Let S1 be the entire root signal set, and S2 be a subset of S1 , where we exclude roots 7 and 4. The reason of excluding roots 7 and 4 is that they are less likely to occur than the rest of the set. In terms of the markoff transition probabilities, the distortion introduced by the filtering process using sets S1 and S2 are

$$D_{f1} = 4\,(2\,P_0 P_1^2 + P_1^3) \ , \text{ and } D_{f2} = 4\,(3\,P_0 P_1^2 + P_1^3).$$

Figure 4 shows the optimum bound R(D) along with the root coding points for the 2 sets. This plane shows that if the image has high correlation, filtering process will modify on the average 1 bit in 4 entire blocks. If the input image has low correlation the expected distortion is about 1 bit per block (Recall there are 16 bits per block).

5. SEQUENTIAL ANALYSIS

In the one dimensional root coding we have taken advantage of the x axis correlation only. By removing some mutual information in the y axis, we can reduce the bit rate further. One approach would be to model the source as a two dimensional markov process and by an analysis similar to the one dimensional case, one can remove the x and y image correlation. Here, we avoid that complicated approach and remove y axis redundancy by a sequential root analysis.

Each block is coded by four consecutive roots; where each root has four samples. Each root is treated as a random variable that can take 10 posible states. The distribuition for this random variable was obtained in the previous section. Denote the root class j and the k th root signal of the block by

$$\Upsilon_k^j \quad , \textit{where} \quad 1 \leqq j \leqq 10 \ , \ 1 \leqq k \leqq 4.$$

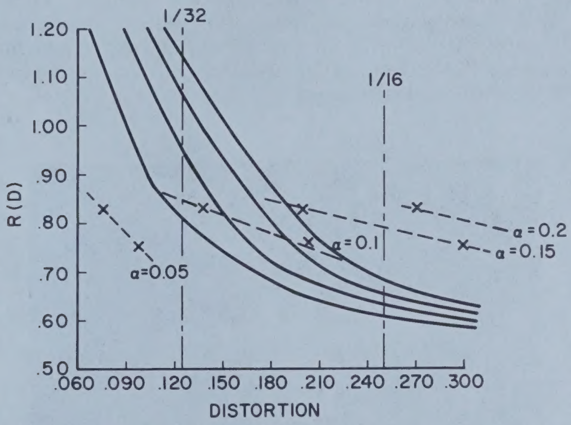


Fig. 4 Rate Distortion Plane.

As mentioned before, each root signal has four components

$$\Upsilon_k^j \equiv (s_n^k , s_{n+1}^k , s_{n+2}^k , s_{n+3}^k),$$

where k denotes the root vertical location within the block, and the lower subscript is the column of the block. The sequential conditional probability of two consecutive roots, given the first one is

$$\mathbf{Pr}(\Upsilon_{k+1}^j / \Upsilon_k^i) = \frac{\mathbf{Pr}(\Upsilon_k^i / \Upsilon_{k+1}^j) \ \mathbf{Pr}(\Upsilon_{k+1}^j)}{\mathbf{Pr}(\Upsilon_k^i)}. \tag{13}$$

Where

$$\mathbf{Pr}(\Upsilon_k^i) = \sum_{j=1}^{10} \mathbf{Pr}(\Upsilon_k^i / \Upsilon_{k+1}^j) \ \mathbf{Pr}(\Upsilon_{k+1}^j) \tag{14}$$

Notice that (14) can be evaluated directly since

$$\mathbf{Pr}(\Upsilon_k^i) \text{ and } \mathbf{Pr}(\Upsilon_{k+1}^j)$$

are given in (5). Define the difference measure

$$d_n = |[s_n^k - s_n^{k+1}]| \ , \text{and } d_T = \sum_{n=1}^{4} d_n. \tag{15}$$

Applying the markoff model in the vertical direction,

$$\mathbf{Pr}(\Upsilon_k^i / \Upsilon_{k+1}^j) = P_0^{4-d_T} \, P_1^{d_T}. \tag{16}$$

It is important to point out that Po and P1 in (16) are the transition probabilities in the vertical direction. In general, the correlation in the horizontal axis is higher than the correlation in the vertical direction. This is so because of the horizontal sampling process. If we ignore this fact in computing (16), values differing up to 20 % of the true transition probabilities can be obtained. Replacing (16) and (14) into (13) we obtain a measure of the mutual information on the y axis. The conditional transition probabilities (13) among all values of the root random variable are easily obtained. The experimental and the theoretical values for the transition probabilities from root 2 are shown in table 2. Note that there are some very unlikely transition paths. By restricting certain transition paths, we can reduce the bit rate with a low probability of distortion. This method of encoding is based on a trellis. The trellis will allow all transitions which have high probability of occurrence. As an example, we will encode an image, were only 4 transitions are allowed. We allow 3 bits to the first row in order to start the state trellis with an exact location, and 2 bits for each transition. Figure 5 shows a BTC coded image and the trellis compressed image, at 16 bits/block and 9 bits/block respectively.

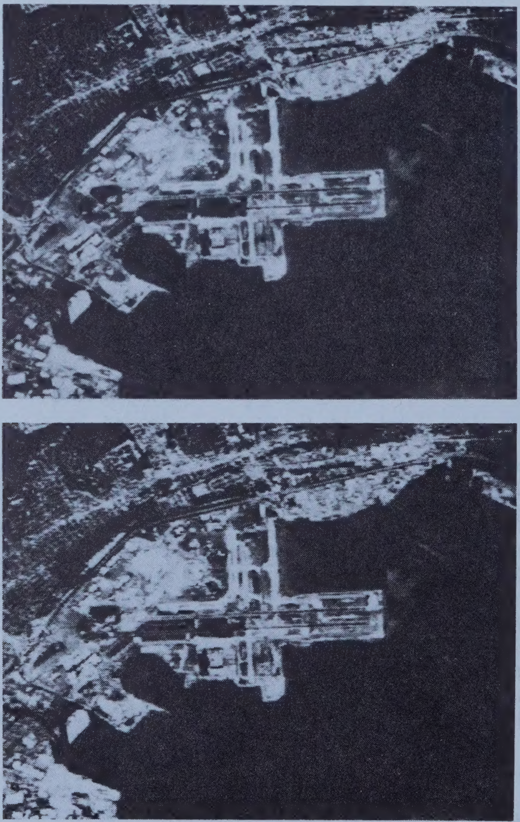


Fig. 5 BTC and TRELLIS Coded Images.

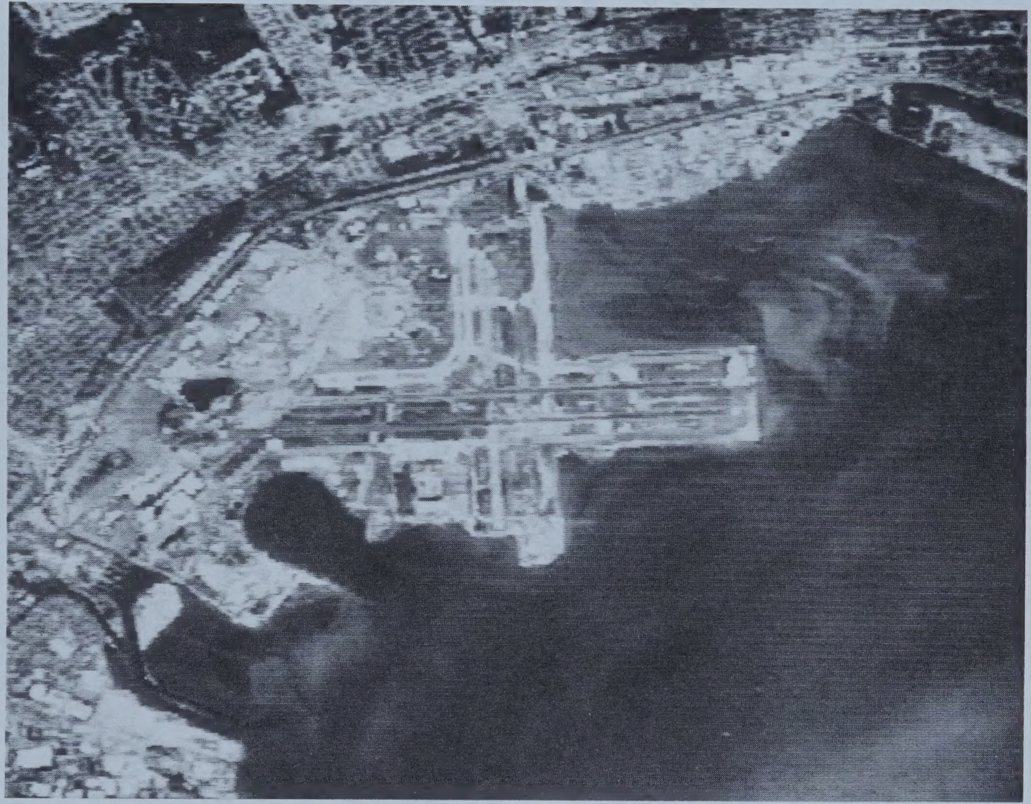


Fig 5 . (cont.) Enlarged Trellis Coded Image.

TABLE 2 Conditional Transition Probabilities.

j	$\Pr\left\{ T_k^j + 1 \big/ T_k^2 \right\}$	Experimental values*
1	0.38	0.22
2	0.32	0.29
3	0.12	0.21
5	0.034	0.08
6	0.034	0.04
8	0.015	0.03
9	0.014	0.02
10	0.06	0.09

* For Fig 5.

6. CONCLUSION

A new method of coding BTC and binary images is presented. This method uses a simple nonlinear digital filter, namely a median filter. The performance of the coding scheme is evaluated in a rate-distortion plane. Finally, a trellis encoder is presented in which we obtain higher data compression at a cost of higher probability of distortion.

REFERENCES

[1] E.J. Delp and O.R. Mitchell. "*Image compression using BTC*". IEEE Trans Comm, vol Com-27. Sept. 1979.

[2] A. Papoulis. *Probability, Random Var. and Stochastic Processes.* McGraw Hill 1965.

[3] T.S. Huang. "*Coding of Two Tone Images*". IEEE Trans. Communications vol COM-25 Nov. 1977.

[4] G. R. Arce and N. C. Gallagher, jr. "*Root Signal Set Analysis for Median Filters*". Proccedings of the 18 th annual Conference on Communications, Control, and Compt. Oct. 1980

[5] S. O. Rice. "*Mathematical Analysis of Random Noise*". Bell System Tech J. Vols. 23 & 24 , 1944 & 1945.

[6] J. Capon. "*A Probabilistic Model for the Runlength Coding of Pictures*". IRE Trans. on Information Theory , Dec. 1959.

[7] J.Viterbi and K. Omura. "*Principles of Digital Communications and Coding*" McGraw Hill 1979.

C-MATRIX TRANSFORM

R. Srinivasan and K. R. Rao
Electrical Engineering Department
The University of Texas at Arlington
P.O. Box 19016
Arlington, TX 76019

SUMMARY

Because of its near optimal performance discrete cosine transform (DCT) has been widely used in digital signal and image processing. Hein and Ahmed [1] have shown that the DCT can be implemented via the Walsh-Hadamard transform (WHT) through a conversion matrix which has block diagonal structure. Jones et al [2] have generalized this process and they have shown that any even/odd transform (This transform has one-half even and one-half odd vectors.) can be expressed in terms of any other even/odd transform and the conversion matrix has sparse block diagonal format. During this development they have obtained an approximation to the DCT for N=8 (N is the length of the data sequence.). This transform called C - matrix transform (CMT) has the following properties:

(i) It is an even/odd transform.
(ii) The conversion matrix has only integers as its elements. Hence, only integer arithmetic is involved.
(iii) Its performance is very close to the DCT.

In the present paper the CMT is extended to N = 16. (The complexity of developing the CMT increases exponentially as N increases.) The effectiveness of the CMT is compared with other transforms based on variance distribution, rate distortion, mean-square error for scalar Wiener filters and computational complexity. The inherent advantage of the CMT appears to be the simplicity in its implementation (integer arithmetic plus adders/subtractors) yet negligible performance degradation compared to the DCT. Hardware architecture and data compression capability of the CMT are developed. It is felt that a 8 or 16-point CMT can be made operational on a single chip for real-time processing of audio and video signals. The lower (8x8) block diagonal matrix for the conversion matrix (The upper (8x8) matrix is described in [2].) is

$$\frac{1}{13}\begin{bmatrix} 12 & 0 & 0 & -1 & 4 & 0 & 2 & 2 \\ 0 & 12 & -1 & 0 & 0 & 4 & -2 & 2 \\ 0 & 1 & 12 & 0 & -2 & 2 & 4 & 0 \\ 1 & 0 & 0 & 12 & -2 & -2 & 0 & 4 \\ -4 & 0 & 2 & 2 & 12 & 0 & 0 & 1 \\ 0 & -4 & -2 & 2 & 0 & 12 & 1 & 0 \\ -2 & 2 & -4 & 0 & 0 & -1 & 12 & 0 \\ -2 & -2 & 0 & -4 & -1 & 0 & 0 & 12 \end{bmatrix}$$

REFERENCES

1. D. Hein & N. Ahmed, "On a real-time Walsh-Hadmard/cosine Transform Image Processor," IEEE Trans. Electromagn. Compat.,Vol. EMC-20, pp. 453-457, Aug. 1978.

2. H. W. Jones, D. N. Hein & S. C. Knauer, "The Karhunen-Loeve, Discrete Cosine and Related Transforms Obtained via the Hadamard Transform," Intl. Telemetering Conf., Los Angeles, CA, Nov. 14-16, 1978.

SYSTOLIC AND SIMD ALGORITHMS FOR DIGITAL FILTERING

MARK A. YODER & LEAH J. SIEGEL
Purdue University
School of Electrical Engineering
West Lafayette, IN 47907

ABSTRACT

Systolic array and SIMD (single instruction stream-multiple data stream) algorithms to perform digital filtering are presented. The machine capabilities required for the systolic and SIMD approaches are compared, and relationships between the algorithms are discussed. Detailed complexity analyses of the parallel algorithms are given.

INTRODUCTION

With the advent of VLSI technology, large-scale parallel processing systems with as many as 2^{14} processors have become feasible [1, 8, 13]. Two approaches to using a large number processors are systolic arrays [6] and SIMD machines [3].

A systolic machine is a network of simple processing elements (PEs) that circulate data in a regular fashion [6]. The network configuration for a systolic machine is particular to the algorithm (or class of algorithms) being implemented. In general, the data flow in a systolic array can be viewed as a multidimensional pipeline.

An SIMD (single instruction stream-multiple data stream) machine [3] typically consists of a control unit, N processing elements (PEs) each a processor with its own memory, and an interconnection network. The control unit broadcasts instructions to all PEs, and each active PE executes the instruction on the data in its own memory. The PEs may be simple (e.g., CLIP4 [2], MPP [1]) or may be complete microprocessors (e.g., PASM [10]).

Both systolic array and SIMD structures are especially suitable for exploiting the parallelism inherent in certain tasks performed on vectors and arrays [e.g., 6, 11, 12]. In this paper, systolic array and SIMD algorithms to perform recursive digital filtering are presented. The basic operations in digital filtering are the computation of sum of products terms, with output y_m given by

$$y_m = \sum_{k=1}^{p} a_k y_{m-k} + \sum_{k=0}^{q} b_k x_{m-k} \qquad p \leq m \leq M \tag{1}$$

where x_m is the input to the filter, the a_k's and b_k's are the filter coefficients, and M is the number of samples in the signal to be filtered.

Three parallel algorithms are presented. The first is a systolic array algorithm, the second is an SIMD implementation derived from this algorithm, and the third is an SIMD algorithm that exploits the full features of an SIMD machine. The algorithms are evaluated in terms of the number of arithmetic operations and interprocessor transfers required. Because of the recursive nature of the computation, the systolic array appears to be a natural structure for implementing the digital filter. In [4], the use of the "skewed" SIMD mode of parallelism to perform digital filtering has been described. In this mode, all PEs execute the same instruction stream, but the executions of a given instruction are skewed in time. The time skew is used to allow different PEs to perform the same instructions on different time samples. The SIMD algorithms presented here demonstrate strategies for handling the recursive computations in a different manner.

This material is based upon work supported by the National Science Foundation under Grant ECS-7909016.

SYSTOLIC ARRAY ALGORITHM

The systolic algorithm presented here combines the finite impulse response (FIR) and recursive filter algorithms in [6] into a general recursive filter algorithm. It is based on a linear array of PEs, with each PE holding one filter coefficient and data flowing in opposite directions in two pipelines. One pipe circulates the input data (x_m values) while the other passes partial results in the y_m computation. Details follow.

Kung's FIR filter algorithm computes a (q+1)-tap FIR filter using a linear array of q+1 systolic PEs. It solves the equation in which y_m is computed using the second summation in equation (1). The output y_m can be computed by the following recurrence relation:

$$
\begin{aligned}
y_m^{(0)} &= 0 \\
y_m^{(k+1)} &= y_m^{(k)} + b_{q-k} x_{m-q+k} \qquad 0 \leq k \leq q \qquad (2) \\
y_m &= y_m^{(q+1)}
\end{aligned}
$$

The above recurrences can be evaluated by pipelining the x_m and $y_m^{(k)}$ values through q+1 linearly connected processors as shown in Fig. 1. Each processor has three registers, R_b, R_x, and R_y, which hold b, x, and y values respectively. Initially, all R_x and R_y registers contain zeros, and the R_b register in processor i contains b_{q-i}. Each cycle of the array consists of the steps shown in Fig. 1. $y_m^{(k)}$ is computed in PE k-1, and the output is produced in PE q. The data flowing up (the x_m values) must be synchronized with the data flowing down (the $y_m^{(k+1)}$ values) so that they meet in the correct PE with the correct coefficient. Therefore during odd numbered cy-

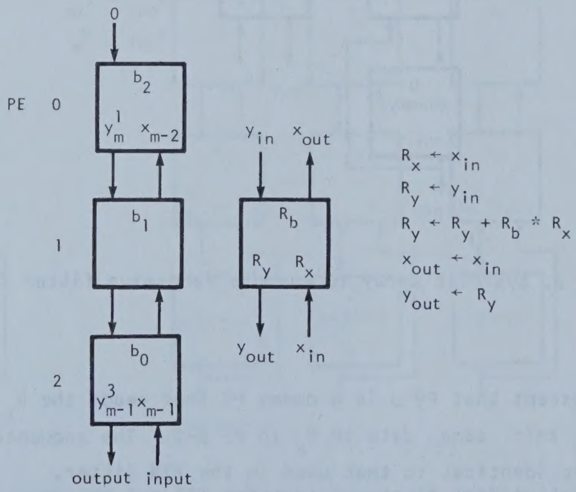


Fig. 1. Systolic array to compute FIR filter for q=2.

cles, only even numbered PEs contain valid data, and during even numbered steps only the odd PEs contain valid data. Thus only half of the PEs are active during a given cycle. One output value is therefore computed every two cycles of the systolic array, where during each cycle, the operations performed are the simultaneous transfer of data in the two pipes, plus the one addition, one multiplication, and one assignment shown in equation (2). (This is essentially the same as Kung's algorithm, except for a few notational differences that will make this algorithm more compatible with the general recursive digital filter algorithm presented later.)

Kung's recursive filter algorithm solves the equation in which y_m is computed using the first summation in equation (1). The algorithm is based on the recurrence relation:

$$y_m^{(0)} = 0$$

$$y_m^{(k+1)} = y_m^{(k)} + a_n y_{m-n} \qquad 0 \leq k \leq p-1 \tag{3}$$

$$y_m = y_m^{(p)}$$

to evaluate y_m where $n=p-k$. The computation is performed using a linear array of p+1 PEs as shown in Fig. 2. Each PE is the same as in the FIR filter

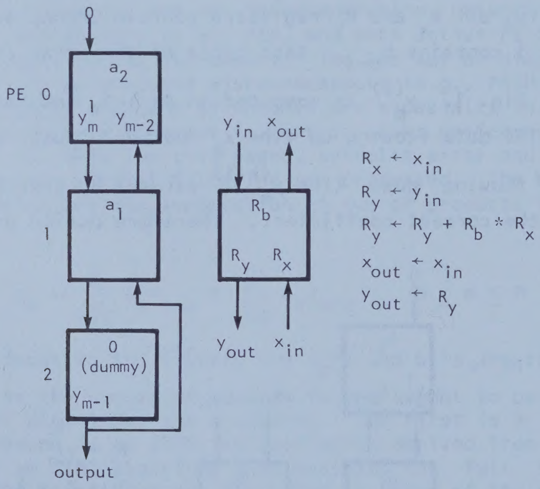


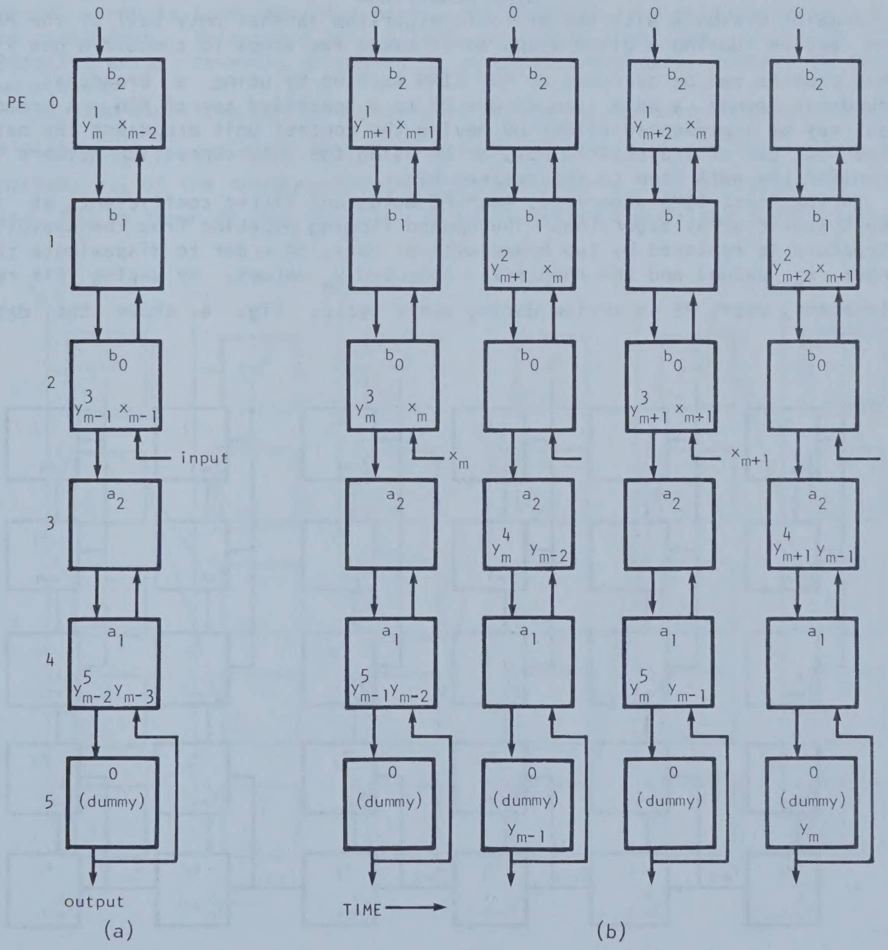
Fig. 2. Systolic array to compute recursive filter for p=2.

algorithm, except that PE p is a dummy PE that reads the R_y data from PE p-1 and routes this same data to R_x in PE p-1. The sequence of steps during each cycle is identical to that used in the FIR filter.

The general digital filter of equation (1) can be computed by combining the two algorithms discussed above. The recurrence relations used are:

$$
\begin{aligned}
y_m^{(0)} &= 0 \\
y_m^{(k+1)} &= y_m^{(k)} + b_{q-k}\,x_{m-q+k} && 0 \le k \le q \\
y_m^{(k+1)} &= y_m^{(k)} + a_n\,x_{m-n} && q+1 \le k \le p+q \\
y_m &= y_m^{(p+q+1)}
\end{aligned}
\tag{4}
$$

The recurrences can be evaluated by pipelining the x_m and $y_m^{(k)}$ values through p+q+2 linearly connected PEs as shown in Fig. 3a. Input x_m is fed into R_x in PE q and output y_m appears in R_y in PE p+q. Fig. 3b is the data flow diagram for the linear array. Each column of the data flow diagram



PE 0
1
2
3
4
5
0
b2
b1
b0
a2
a1
0 (dummy)
1 ym xm-2
3 ym-1 xm-1
5 ym-2 ym-3
input
output
(a)
1 ym+1 xm-1
3 ym xm
xm
5 ym-1 ym-2
2 ym+1 xm
4 ym ym-2
ym-1
1 ym+2 xm
3 ym+1 xm+1
xm+1
5 ym ym-1
2 ym+2 xm+1
4 ym+1 ym-1
ym
TIME
(b)

Fig. 3a) Systolic array to compute general digital filter for p=2, q=2.
b) Data flow diagram for (a).

represents the contents of each register in each PE after a given cycle. Moving from left to right shows how the data changes from one cycle to the next. The arrows show where R_x and R_y will be transferred on the next cycle. As in the component algorithms, only half of the PEs are active during a given cycle. Before the first cycle, the correct coefficient is loaded into each PE, and the R_x and R_y registers are set to zero. The first q cycles shift x_0 from the input line in PE q to R_x in PE 0, x_1 from PE q to R_x in PE 2, ..., and $x_{\lfloor q/2 \rfloor}$ to R_x in PE $2\lfloor q/2 \rfloor$ (i.e., initializing the array by placing the first $\lfloor q/2 \rfloor+1$ input values in every other PE, starting with x_0 in PE 0). After p+q+1 more cycles, one output value is computed every two cycles of the systolic array, where during each cycle, the operations performed are the simultaneous transfer of data in the two pipes, plus one addition, one multiplication, and one assignment.

SIMD ALGORITHMS

A major drawback with the systolic algorithm is that only half of the PEs are active during a given step, so it takes two steps to compute a new y_m. This problem can be overcome on the SIMD machine by using a broadcast. A broadcast sends a data item in one PE to a specified set of PEs. A broadcast may be implemented either by having the control unit broadcast the data item to all of the desired PEs, or by using the interconnection network to transfer the data item to the desired PEs.

In the first SIMD algorithm, each PE holds one filter coefficient, as in the systolic array algorithm. The upward flowing pipeline from the systolic structure is replaced by two broadcasts of data, in order to disseminate the input x_m values and the completely computed y_m values. By making this replacement, every PE is active during every cycle. Fig. 4 shows the data

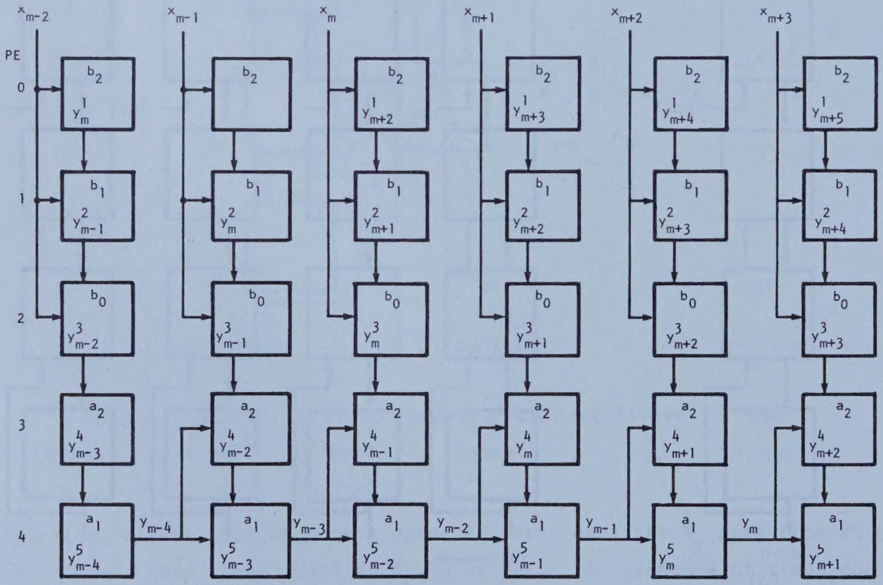

x_{m-2}
x_{m-1}
x_m
x_{m+1}
x_{m+2}
x_{m+3}
PE
0
1
2
3
4
b_2
y^1_m
b_1
y^2_{m-1}
b_0
y^3_{m-2}
a_2
y^4_{m-3}
a_1
y^5_{m-4}
y_{m-4}
b_2
b_1
y^2_m
b_0
y^3_{m-1}
a_2
y^4_{m-2}
a_1
y^5_{m-3}
y_{m-3}
b_2
y^1_{m+2}
b_1
y^2_{m+1}
b_0
y^3_m
a_2
y^4_{m-1}
a_1
y^5_{m-2}
y_{m-2}
b_2
y^1_{m+3}
b_1
y^2_{m+2}
b_0
y^3_{m+1}
a_2
y^4_m
a_1
y^5_{m-1}
y_{m-1}
b_2
y^1_{m+4}
b_1
y^2_{m+3}
b_0
y^3_{m+2}
a_2
y^4_{m+1}
a_1
y^5_m
y_m
b_2
y^1_{m+5}
b_1
y^2_{m+4}
b_0
y^3_{m+3}
a_2
y^4_{m+2}
a_1
y^5_{m+1}

Fig. 4. Data flow diagram for SIMD general digital filtering algorithm for p=2, q=2.

flow diagram using this technique. As each y shifts into a given PE, the correct coefficient and x (in PEs 0 through p) or y (in PEs p+1 through p+q+1) are there to meet it. Moreover, if a given PE receives an x as a result of the broadcast, it will not receive a y as a result of the broadcast, and vice versa. If the interconnection network rather than the control unit is used to perform the broadcasts, it may be possible to perform the two broadcasts to disjoint sets of PEs simultaneously [7]. Whether or not this is possible will depend on factors such as the type of interconnection network used, the actual sets to which the data items are being broadcast, and the way in which the x values enter the system. The data flow of the partial results (the $y_m^{(k+1)}$ values) and the placement of one coefficient per PE is the same as the systolic array algorithm. The replacement of the upward pipe by two broadcasts simplifies the synchronization problems, and allows all PEs to be active at every step. One output value is produced on every cycle of the SIMD algorithm, where during each cycle, the operations performed are the two possibly simultaneous broadcasts, one data transfer of partial results (the remaining pipe), one addition, one multiplication, and one assignment. This cycle will clearly be longer than the cycle in the systolic array, however, an output is produced every cycle instead of every two cycles.

The SIMD algorithm can be improved by arranging the data so that partial results ($y_m^{(k+1)}$ values) do not have to be shifted from one PE to another. Instead, all of the steps needed to compute a given y_m are performed in the same PE, as shown in Fig. 5. Each PE holds all of the filter coefficients,

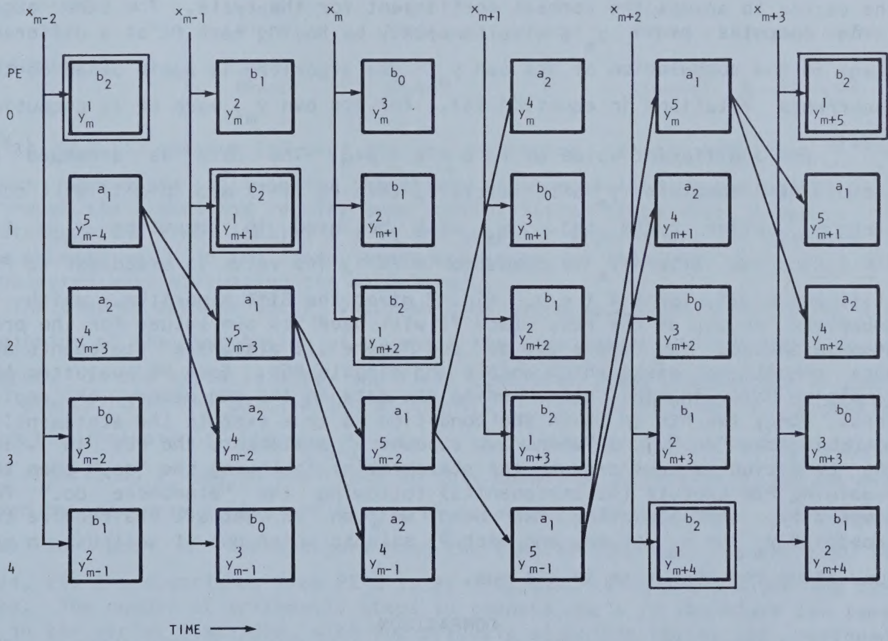

x_{m-2}
x_{m-1}
x_m
x_{m+1}
x_{m+2}
x_{m+3}
PE
0
1
2
3
4
b_2 y^1_m
a_1 y^5_{m-4}
a_2 y^4_{m-3}
b_0 y^3_{m-2}
b_1 y^2_{m-1}
b_1 y^2_m
b_2 y^1_{m+1}
a_1 y^5_{m-3}
a_2 y^4_{m-2}
b_0 y^3_{m-1}
b_0 y^3_m
b_1 y^2_{m+1}
b_2 y^1_{m+2}
a_1 y^5_{m-2}
a_2 y^4_{m-1}
a_2 y^4_m
b_0 y^3_{m+1}
b_1 y^2_{m+2}
b_2 y^1_{m+3}
a_1 y^5_{m-1}
a_1 y^5_m
a_2 y^4_{m+1}
b_0 y^3_{m+2}
b_1 y^2_{m+3}
b_2 y^1_{m+4}
b_2 y^1_{m+5}
a_1 y^5_{m+1}
a_2 y^4_{m+2}
b_0 y^3_{m+3}
b_1 y^2_{m+4}
TIME ⟶

Fig. 5. Data flow diagram for improved SIMD general digital filtering algorithm for p=2, q=3. The double boxes indicate the start of a new y_m computation.

PE	COEF[0]	COEF[1]	COEF[2]	....	COEF[p+q-1]	COEF[p+q]
0	b_q	b_{q-1}	b_{q-2}	....	a_2	a_1
1	a_1	b_q	b_{q-1}		a_3	a_2
2	a_2	a_1	b_q		a_4	a_3
.	.	.	.		.	.
p	a_p	a_{p-1}	a_{p-2}		b_1	b_0
p+1	b_0	a_p	a_{p-1}		b_2	b_1
.	.	.	.		.	.
q+p-1	b_{q-2}	b_{q-3}	b_{q-4}	....	b_q	b_{q-1}
q+p	b_{q-1}	b_{q-2}	b_{q-3}	....	a_1	b_q

Fig. 6. Skewed coefficient storage for SIMD algorithm.

and uses an indexing operation to select which coefficient to use at a given step of the algorithm. Partial results are accumulated within the PEs, rather than being pipelined through them. The data transfers required are two (possibly simultaneous) broadcasts, one of the current input signal value, and one of a completed output value. All PEs are always active, and one output is completed each cycle of the algorithm, where the computations during a cycle are one indexing operation to select a filter coefficient, two broadcasts, one addition, one multiplication, and one assignment. For this SIMD algorithm, the coefficients are arranged as a vector in each PE as shown in Fig. 6. This arrangement allows each PE to use the same index into the vector to access the correct coefficient for the cycle. The SIMD algorithm computes p+q+1 y_m's simultaneously by having each PE at a different stage in the computation of its own y_m. The algorithm is again based on the recurrence relations in equation (4). For its own y_m, each PE is computing $y_m^{(k+1)}$ for a different value of k, $0 \le k \le p+q$. The data is arranged so that if PE i completes y_m after cycle t, then PE (i+1) mod (p+q+1) will complete y_{m+1} after cycle t+1. y_m will be used in computing y_{m+j} for $1 \le j \le p$, so after y_m is computed in PE i, its value is broadcast to PEs (i+j) mod p+q+1, for $1 \le j \le p$. Fig. 7 gives the SIMD algorithm, which is broadcast to all of the PEs. Each PE will have its own values for the program variables. The "where ... do" and "where ... elsewhere" statements are data conditional masks which enable and disable PEs. Each PE evaluates the condition following the "where" using the data in its own memory or registers. Only the PEs in which the condition is true execute the statement(s) following the "do." In a "where ... elsewhere" statement, the PEs in which the condition is true execute the statement(s) following the "do," then the remaining PEs execute the statement(s) following the "elsewhere do." For simplicity, the algorithm has been written so that all PEs receive the broadcast y_m and x_m values, and each PE selects which one it will use in accumulating the next term in its sum.

COMPARISON

Table 1 shows the times for the serial and parallel algorithms. The parallel algorithms can be compared in three ways: (1) total time to compute one y_m, (2) number of y_m's computed per unit time (throughput) (the throughput can also be considered by measuring the time between successive

```
ADDR     = Address of PE (e.g., ADDR = 0 in PE 0)
DTRin    = Data Transfer Register input to interconnection network
DTRout   = Data Transfer Register output from interconnection network
COEF[]   = Vector of coefficients (see Fig. 6)
FLAG[i]  = Equals 1 if COEF[i] is an "a" coefficient
SUM      = Contains partially computed $y_m$

for j <- 0 to M+p+q-i do
        where ADDR = j mod p+q+i do    /* select the PE containing the newly */
             DTRin <- SUM;             /* completed y value: $y_{j-(q+1)}$. */
             SUM <- 0;                 /* Broadcast that y value and */
             broadcast;                /* start a new sum in that PE */
        where FLAG[j mod p+q+1] = 1 do /* In each PE, select to use */
             TMP <- DTRout;            /* either the broadcast y value */
        elsewhere do
             TMP <- $x_j$;                /* or the new x value */

        SUM <- SUM + TMP * COEF[j mod p+q+1];
```

Fig. 7. SIMD general digital filtering algorithm.

Table 1. Execution times for serial, systolic, and SIMD algorithms.

	output(s)	additions	multiplications	shifts	broadcasts	speed up
serial:	1	p+q+1	p+q+1	0	0	
	M	M(p+q+1)	M(p+q+1)	0	0	
systolic:	1	p+q+1	p+q+1	p+q+1	0	
	M	2(M-1)+p+q+1	2(M-1)+p+q+1	2(M-1)+p+q+1	0	(p+q+1)/3
SIMD(1):	1	p+q+1	p+q+1	p+q+1	2(p+q+1)	
	M	M+p+q	M+p+q	M+p+q	2(M+p+q)	2(p+q+1)/(2+3k)
SIMD(2):	1	p+q+1	p+q+1	0	2(p+q+1)	
	M	M+p+q	M+p+q	0	2(M+p+q)	(p+q+1)/(1+k)

y_m's), and (3) speedup [5] over the corresponding serial algorithm. The times considered will be for the steady state operation of the algorithms. Although the algorithms require some initialization steps (for example, to distribute the first $\lfloor q/2 \rfloor+1$ x's in the systolic algorithm), the majority of the processing is in the steady state operation, so this is what will be considered when evaluating the algorithms.

The time to compute one y_m value will be taken to be the time from the beginning of the computation of y_m until the time that y_m is available as an output. In the systolic array and first SIMD algorithm, computation of each y_m starts with the calculation of the $b_q x_{m-q}$ term in PE 0 and is completed upon the inclusion of the $a_1 y_{m-1}$ term in the sum of PE p+q. (In the systolic algorithm, y_m may be available at this point, or access to y_m may be delayed by one array cycle, until y_m arrives at the output line in the dummy PE.) For both of these algorithms, the time to compute y_m is the time to move, via the algorithm, from PE 0 to PE p+q, comprising p+q+1 algorithm cycles. The number of arithmetic steps to compute one y is therefore the same as in the serial algorithm, with the systolic algorithm having an overhead of p+q+1 shifts and the first SIMD algorithm having an overhead of p+q+1 shifts and 2(p+q+1) broadcasts. (Throughout this section, it is assumed that the SIMD broadcasts do not occur simultaneously.) In the second SIMD algorithm, the time to compute one y is the time for a single PE to perform the arithmetic operations (i.e., the serial time) plus the time for p+q+1 broadcasts of x values and p+q+1 broadcasts of completed y values. As in the first SIMD algorithm, broadcasts of x_{m+1} through x_{m+q} and of y_{m-q-p} through y_{m-q-1} contribute to the time to compute y_m, even though they are not used in the y_m calculations.

For all of the parallel algorithms, the time to compute M output values will be time to compute one y value plus the time to compute (M-1) subsequent y values. The latter time can be obtained by considering the time between successive y values. The time between successive y's in the systolic array is two additions, multiplications, and shifts since one y is computed every two cycles. The SIMD algorithms on the other hand perform one addition, one multiplication, and either two broadcasts and one shift for SIMD (1) or two broadcasts for SIMD (2) between successive y values, since they compute one y every cycle. Depending on the SIMD broadcast vs. systolic shift time, the second SIMD algorithm may therefore have a greater overall throughput.

The speedup of an algorithm is (serial time/parallel time) [5]. Assume that additions and multiplications take one time unit on all machines, and data transfers (shifts or broadcasts) require one time unit on the systolic array and k units on the SIMD machine. The value of k will depend on a number of factors, including implementation details of the systolic and SIMD machines. Under the assumption that M >> p+q, the speedups for the parallel algorithms are shown in Table 1. If k=2, SIMD (2) will have the same speedup as the systolic array. If a multistage interconnection network such as the multistage cube [9] or augmented data manipulator [7] is used to perform the broadcasts, it is very unlikely that $k \leq 2$. Unless the broadcasts can be performed simultaneously, the speedup for the systolic array will be significantly greater than for the SIMD algorithm. However, if the broadcasts are performed by the control unit, smaller values for k may be feasible, and the systolic and SIMD algorithms may have comparable speedups.

The systolic and SIMD algorithms can also be compared with respect to the ease with which the machine size/problem size relationship can be changed. In particular, assume that an order p+q digital filter has been implemented using the above techniques. Consider the impact of deciding to use a higher order filter. Let the new filter have p'+q'+1 coefficients, where p'+q'+1 > p+q+1. With some alterations, the second SIMD algorithm can implement a filter having p'+q'+1 coefficients with fewer than p'+q'+1 PEs. In the original algorithm, where a p+q+1 coefficient filter was implemented using p+q+1 PEs, output y_m was computed in PE m mod (p+q+1). Since each y_m computation required p+q+1 cycles, PE m mod (p+q+1) completed computation of y_m at the same time that computation of $y_{m+p+q+1}$ was to be started. To implement a p'+q'+1 coefficient filter with N < p'+q'+1 PEs, each PE again performs all of the computations for a given output. Output y_m is computed in PE m mod N. However, since the number of cycles to compute y_m is greater than the number of PEs, computation of y_{m+N} will not begin until y_m has been completed. Therefore, following every set of N cycles during which y values are started, there will p'+q'+1-N cycles during which no new y computations are started, and following every set of N cycles during which y values are completed, there will be p'+q'+1-N cycles during which no new y values are completed. Execution time will be p'+q'+1 cycles to compute one y value, and $\lceil M/N \rceil$(p'+q'+1)+((M-1) mod N) cycles to compute M y values. For large M, if N = (p'+q'+1)/r for r > 1, then the throughput of the N-PE algorithm will be reduced by approximately a factor of r from that of the (p'+q'+1)-PE algorithm. In contrast, a major hardware modification (adding more registers to add additional coefficients and y_m values) is needed for systolic arrays to handle a larger problem size. It is generally easier to add more PEs to the array than to modify the exsisting PEs. Therefore a systolic array of size p+q+1 can not easily implement a larger problem size.

In summary, synchronous parallel structures for implementing digital filters have been presented. Although exact timing will be implementation dependent, it is most likely that the systolic array approach will be faster than the SIMD algorithms. On the other hand, the SIMD system can accommodate changes in the order of the filter, whereas the systolic array requires hardware modification to handle a change in problem size. Both yield significant speedups over serial processing. For a given application or environment, the choice of structure can be made by considering the relative importance of speed and flexibility.

REFERENCES

[1] K. Batcher, "MPP -- a massively parallel processor," 1979 Int. Conf. Parallel Processing, Aug. 1979, p. 249.

[2] M. J. B. Duff, "Parallel algorithms and their influence on the specification of application problems," in Multi-Computer Algorithms and Image Processing, K. Preston and L. Uhr, eds., Academic Press, New York, NY, 1981.

[3] M. J. Flynn, "Very high-speed computing systems," Proc. IEEE, Vol. 54, Dec. 1966, pp. 1901-1909.

[4] C. J. M. Hodges, T. P. Barnwell, III and D. McWhorter, "The implementation of an all digital speech synthesizer using a multimicroprocessor architecture," 1980 IEEE Int. Conf. Acoust., Speech, Signal Proc., Apr. 1980, pp. 855-858.

[5] D. J. Kuck, "A survey of parallel machine organization and programming," Computing Surveys, Vol. 9, Mar. 1977, pp. 29-59.

[6] H. T. Kung, "The structure of parallel algorithms," in Advances in Computers, Vol. 19, M. C. Yovits, ed., Academic Press, New York, NY, 1980.

[7] R. J. McMillen and H. J. Siegel, "Dynamic rerouting tag schemes for the augmented data manipulator network," 8th Symp. Comp. Arch., May 1981, pp. 505-516.

[8] M. C. Pease, "The indirect binary n-cube microprocessor array," IEEE Trans. Comp., Vol. C-26, May 1977, pp. 458-573.

[9] H. J. Siegel and R. J. McMillen, "The multistage cube: a versatile interconnection network," Computer, to appear.

[10] H. J. Siegel, L. J. Siegel, F. C. Kemmerer, P. T. Mueller, Jr., H. E. Smalley, Jr., and S. D. Smith, "PASM: a partitionable SIMD/MIMD system for image processing and pattern recognition," IEEE Trans. Comp., to appear in Dec. 1981.

[11] L. J. Siegel, P. T. Mueller, Jr., and H. J. Siegel, "FFT algorithms for SIMD machines," 17th Allerton Conf. Communication, Control, and Computing, Oct. 1979, pp. 1006-1015.

[12] L. J. Siegel, H. J. Siegel, and A. E. Feather, "Parallel image correlation," 1981 Int. Conf. Parallel Processing, Aug. 1981, pp. 190-198.

[13] H. Sullivan, T. R. Bashkow, and K. Klappholz, "A large scale homogeneous, fully distributed parallel machine," 4th Symp. Comp. Arch., Mar. 1977, pp. 105-124.

ESTIMATING GENERAL TWO-DIMENSIONAL MOTION FROM IMAGE SEQUENCES

Y.P. HSU
Chinese Academy of Sciences
Peking, China

T.S. HUANG
Coordinated Science Laboratory
University of Illinois
Urbana, Illinois

SUMMARY

An equation for estimating general two-dimensional motion from image sequences is derived, the unknowns being the translations in the x- and the y- directions, the rotation angle, the scaling factor and the coordinates of the rotation and the scaling centers. This nonlinear equation can be decomposed into two sets of linear equations. It is clear from these equations that for arbitrary motion, the unknowns cannot be determined uniquely. However, the unique determination of these parameters becomes possible if we assume simple models for the motion. In particular, if we assume (i) rigid body motion, (ii) center of rotation undergoing constant translation and (iii) scaling due to camera zooming, then the unknowns can be determined uniquely from three consecutive image frames. Our theoretical result was confirmed by computer simulation experiments.

The basis equation for estimating two-dimensional motion (assuming small motion) is

$$\Delta f = -\left(\frac{\partial f}{\partial x}\Delta x_T + \frac{\partial f}{\partial y}\Delta y_T\right) - \left(\frac{\partial f}{\partial x}(y - y_R) - \frac{\partial f}{\partial y}(x - x_R)\right)\theta$$

$$- \left(\frac{\partial f}{\partial x}(x - x_s) + \frac{\partial f}{\partial y}(y - y_s)\right) S_f \qquad (1)$$

where $f(x,y)$ = Intensity at picture element (x,y)

$\Delta f(x,y)$ = Frame difference at (x,y)

$\left(\frac{\partial f}{\partial x},\frac{\partial f}{\partial y}\right)$ = Spatial gradient at (x,y) in the second image frame

$(\Delta x_T,\Delta y_T)$ = Translation

(x_R,y_R) = Rotation center

θ = Angle of rotation

(x_s,y_s) = Scaling center

S_f = Scaling factor

BLOCK TRUNCATION CODING ON PASM

LEAH J. SIEGEL **
EDWARD J. DELP *
T. N. MUDGE *
HOWARD JAY SIEGEL **

* University of Michigan Department of Electrical and Computer Engineering Ann Arbor, MI 48109

** Purdue University School of Electrical Engineering West Lafayette, IN 47907

ABSTRACT

Image data compression techniques are an important class of image processing algorithms. Data compression techniques are used in both the efficient storage and transmission of pictures. The implementation of an image coding algorithm known as Block Truncation Coding (BTC) on the PASM parallel processing system is presented. Two forms of BTC are considered, one where the output of the encoder is fixed length and another where the output is variable length. Both the SIMD and MIMD modes of parallelism are considered. The serial and parallel complexities of the coding algorithms are analyzed and compared. The potential for using a parallel processor to meet real-time constraints in image data compression is illustrated.

I. INTRODUCTION

Image data compression is an important discipline that is finding growing application in such diverse areas as video data transmission, the archival storage of images, the storage and transmission of weather satellite data, and remotely piloted vehicle imaging. Characteristic of image data compression problems are large data sets and a large number of arithmetic operations that in principle can be performed concurrently. In many applications real-time constraints require very high processing rates when coding (decoding) an image for data compression (reconstruction). In view of the potential concurrency in the coding problem, using a parallel processing computer having a large number of processors appears to be the most natural solution. Traditionally, the cost of such machines has prohibited their consideration, however, recent advances in VLSI circuit technology have made such architectures feasible. In paper the detailed design of a data compression algorithm to perform Block Truncation Coding (BTC) for a parallel processing computer, PASM, is presented. The rate at which BTC can be performed is shown to increase with the number of processors in approximately a linear fashion. Thus any real-time constraints can be met if a sufficient number of processors are available.

II. BLOCK TRUNCATION CODING

BTC [1,2] is a data compression scheme based on applying a block adaptive two-level (one bit) moment-preserving quantizer to image data [3]. BTC has proved to be very competitive with classical transform coding techniques, such as the Chen and Smith cosine transform algorithm, as far as image quality [4,5]. The basic BTC algorithm requantizes $n x n$ nonoverlapping blocks of pixels in an image into two gray levels. These gray levels are chosen such that the sample

---

This research was supported by the National Science Foundation under Grants ECS-7909016 and MCS-8009315, by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant No. AFOSR-78-3581, and by the Defense Mapping Agency, monitored by the United States Air Force Rome Air Development Center Information Sciences Division, under Contact No. F30602-78-C-0025 through the University of Michigan. The United States Government is authorized to reproduce and distribute reprints for governmental purposes not-withstanding any copyright notation hereon.

mean and mean square value (or sample standard deviation) are identical with the original nxn block.

One proceeds by first dividing the original picture into nxn blocks (we have used $n=4$ for our examples). Blocks are the coded individually, each into a two level signal. The levels for each block are chosen such that the first two sample moments are preserved. Let $k=n^2$ and let $x_1, x_2, \ldots, x_k$ be the values of the pixels in a block of the original picture. Let

$$\bar{m}_1 = \frac{1}{k} \sum_{i=1}^{k} x_i \qquad (1)$$

be the first sample moment

$$\bar{m}_2 = \frac{1}{k} \sum_{i=1}^{k} x_i^2$$

be the second sample moment and let

$$\bar{\sigma}^2 = \bar{m}_2 - \bar{m}_1^2$$

be the sample variance.

As with the design of any one bit quantizer, we need to find a threshold and two output levels for the quantizer. We have chosen the sample mean as the threshold for this application. Other choices of thresholds are discussed in [1,2]. Therefore if

$$x_i \geq \bar{m}_1 \quad \text{output} = y_2 \qquad (2)$$

or

$$x_i < \bar{m}_1 \quad \text{output} = y_1$$

$$\text{for } i = 1, \ldots. k.$$

where

y_1 and y_2 are the low and high

output levels, respectively. The output levels y_1 and y_2 for a two-level non-parametric moment preserving quantizer are found by solving the following equations:

Let q = number of x_i's greater than $\bar{m}_1$

We then have

$$k\bar{m}_1 = (k-q)y_1 + qy_2 \qquad (3)$$

$$k\bar{m}_2 = (k-q)y_1^2 + qy_2^2$$

Equation 3 is readily solved for y_1 and y_2:

$$y_1 = \bar{m}_1 - \bar{\sigma} \left[\frac{q}{k-q} \right]^{\frac{1}{2}} \qquad (4)$$

$$y_2 = \overline{m}_1 + \overline{\sigma} \left[\frac{k-q}{q} \right]^{\frac{1}{2}}$$

Each block is the described by $\overline{m}_1$, $\overline{\sigma}$, and an nxn bit plane consisting of 1's and 0's depending on whether a given pixel is above or below $\overline{m}_1$. The receiver (decoder) reconstructs the image block by calculating y_1 and y_2 from Equation 4 and placing those values in accordance with the bits in the bit plane.

For each 4x4 block the output of the encoder consists of the quantized sample mean and sample standard deviation and a 16 bit "bit plane" (represented by 1 bit/pixel). The sample mean and sample standard deviation are *jointly* quantized to 10 bits using an empirically derived quantization scheme detailed in [2]. This scheme is based on the fact that if $\overline{m}_1$ is very small or very large then $\overline{\sigma}$ will be small and hence we can assign fewer bits to $\overline{\sigma}$. Therefore, the original 16 pixels are represented by 26 bits or 1.625 bits/pixel compared to 8 bits/pixel in the uncoded state. A variable length encoding extension of BTC exists which can yield a further compression by representing those blocks where the sample standard deviation is zero (or very small) by *only* the sample mean quantized to 8 bits. This variation on the BTC algorithm allows the average data rate to be reduced to a value between 0.5 bits/pixel and 1.625 bits/pixel. The exact rate with depend upon the number of blocks in a given image that meet the small variance condition. In empirical tests with high resolution aerial reconnaissance images the percentage of blocks meeting this condition were about 10%. For "head and shoulder" face images the number of blocks meeting the small variance condition can be as high as 30%. We are of course paying for this reduction in data rate by having the output of the encoder for each block be sometimes 26 bits and at other times 8 bits. This can cause problems when channel noise is present.

The BTC decoding algorithm consists of taking the bit plane for each block and substituting for those pixels greater than the sample mean (as given by the bit plane) the gray level value, y_2, and substituting for those less than the sample mean the gray level value, y_2, such that the sample mean and sample standard deviation are the same as the original block of pixels [1]. The reconstructed image appears slightly enhanced but does not look block-like, as would be expected, because the brightness has been preserved.

Let us quickly review the basic BTC encoding algorithm:

(1) The image is divided in small non-overlapping blocks such as 4 x 4.

(2) The first and second sample moments are computed.

(3) A bit plane is constructed such that each pixel location is coded as "one" or a "zero" depending on whether that pixel is greater than $\overline{m}_1$.

(4) The bit plane, $\overline{m}_1$, and $\overline{\sigma}$ are sent to the receiver.

(5) The picture block is reconstructed such that $\overline{m}_1$ and $\overline{\sigma}$ (alternatively $\overline{m}_2$) are preserved. That is pixels in the bit plane that are "0" are set to "y_1" and the "1"'s are set to "y_2" as in Equation 4. For example, suppose a 4 x 4 picture block is given by the following:

$$x_j = \begin{matrix} 121 & 114 & 56 & 47 \\ 37 & 200 & 247 & 255 \\ 16 & 0 & 12 & 169 \\ 43 & 5 & 7 & 251 \end{matrix}$$

so

$$\overline{m}_1 = 98.75$$

$$\bar{\sigma} = 92.95$$

$$q = 7$$

and

$$y_1 = 16.7 = 17$$

$$y_2 = 204.2 = 204$$

the bit plane is:

$$\begin{matrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{matrix}$$

The reconstructed block becomes:

$$\begin{matrix} 204 & 204 & 17 & 17 \\ 17 & 204 & 204 & 204 \\ 17 & 17 & 17 & 204 \\ 17 & 17 & 17 & 204 \end{matrix}$$

and the sample mean and variance are preserved.

BTC has been used successfully to code various types of pictures including aerial reconnaissance images, multilevel graphics, and video signals using a multitemporal extension of BTC. A variation of BTC has also been used in speech coding at low bit rates.

III. PASM OVERVIEW

Two types of parallel processing systems are SIMD and MIMD. SIMD (single instruction stream-multiple data stream) machines [7], such ad Illiac IV [8] and STARAN [9], typically consist of a set of N processors, N memories, an interconnection network, an a control unit. The control unit broadcasts instructions to the processors, and all enabled ("turned on") processors execute the same instruction at the same time. Each processor executes instructions on a separate data stream. The interconnection network allows interprocessor communication. An MIMD (multiple instruction stream - multiple data stream) machine [7] also typically consists of N processors, N memories, and an interconnection network, but each processor can follow an independent instruction stream (e.g., C.mmp [10] and Cm* [11]). As with SIMD architectures, there is a multiple data stream. PASM is a partitionable SIMD/MIMD multimicroprocessor system being designed for image processing and pattern recognition applications [6]. It will consist of N processors which can be structured as one or more independent SIMD and/or MIMD machines, where N is a power of two and may be as large as 1024.

Figure 1 is a block diagram overview of the PASM system. The heart of the PASM system is the Parallel Computation Unit (PCU), which contains N processors, N memory modules, and an interconnection network. The Micro Controllers are a set of microprocessors which act as the control units for the PCU processors in SIMD mode and orchestrate the activities of the PCU processors in MIMD mode. Control Storage contains the programs for the Micro Controllers. The Memory Management System controls the loading and unloading of the PCU memory modules. The Memory Storage System stores these files. It consists of

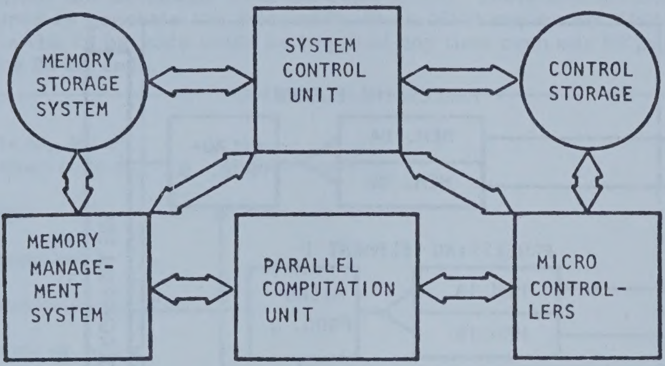


Figure 1. Block Diagram Overview of PASM.

multiple secondary storage devices connected in a fashion that will allow parallel loading/unloading. The System Control Unit is a conventional machine, such as a PDP-11, and is responsible for the overall coordination of the activities of the other components of PASM.

The organization of the PCU is shown in Figure 2. The PCU processors are microprocessors that perform the actual SIMD and MIMD computations. The PCU memory modules are used by the PCU processors for data storage in SIMD mode and both data and instruction storage in MIMD mode. A pair of memory units is used for each PCU memory module so that data can be moved between one memory unit and secondary storage while the PCU processor operates on data in the other memory unit (double buffering). A processor and its associated memory module are termed a processing element (PE). The interconnection network provides a means of communication among the PCU's PE's. Either the Augmented Data Manipulator network [12] or the Generalized Cube Network [13] will be used in PASM.

This brief summary of the PASM organization is provided as background for the following sections. For further details of the system architecture, see the references indicated.

IV. PARALLEL BLOCK TRUNCATION CODING

Coding

Parallel implementation of BTC on PASM is divided into two cases. The fixed bit rate method is structured as an SIMD process. For the variable rate method, both SIMD and MIMD implementations are examined.

For data compression (coding) under both methods, an image having $4 \lambda N$ lines (λN "line groups"), $\lambda \leq 1$, will be allocated to the PEs such that PE i holds line groups λi through $\lambda (i+1)-1$, $0 < i < N$. Each PE will therefore hold complete 4x4 blocks, and will not need data from any other PEs. For smaller images (number of line groups < N), the most efficient approach to coding will be to use the partitionability of PASM, allowing several images to be processed in parallel. The partition sizes will be chosen so that each PE holds at least one line group. In both cases (number of line groups < or > N), the double buffering of the PASM

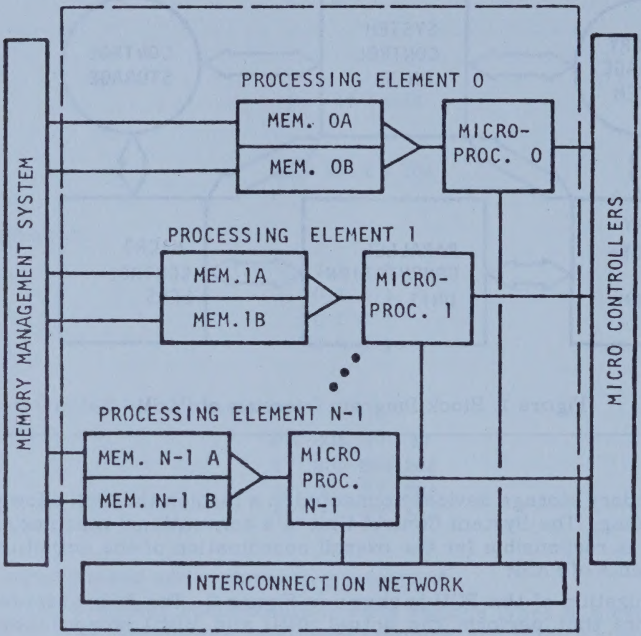

PROCESSING ELEMENT 0
MEM. 0A
MEM. 0B
MICRO-PROC. 0
PROCESSING ELEMENT 1
MEM.1A
MEM.1B
MICRO-PROC. 1
PROCESSING ELEMENT N-1
MEM. N-1 A
MEM. N-1 B
MICRO PROC. N-1
MEMORY MANAGEMENT SYSTEM
MICRO CONTROLLERS
INTERCONNECTION NETWORK

Figure 2. PASM Parallel Computation Unit.

memories will be used so that while one image is being processed, another can be obtained from the Memory Storage System.

For fixed rate coding PASM will operate in SIMD mode with each PE simply performing the BTC algorithm on its portion ("stripe") of the image. That is, all PEs will execute the same BTC algorithm in lockstep, following the same instruction stream; each PE will process a different stripe of the image. Because all PEs will code the same number of blocks and because no inter-PE communications are needed, the execution time for the SIMD algorithm on an N-PE machine will be 1/N-th that of the serial algorithm.

For the variable rate scheme, more operations are required to code a block to 26 bits than to 8 bits, so different PEs may not be performing the same coding operations. Figure 3 shows the flow of computations for the variable rate coding. The number of operations performed to generate the 8-bit code is a basically subset of the operations to generate the 26-bit code. In an SIMD machine each PE would execute the operations appropriate for the block which it is coding. This could be implemented using data conditional masks to enable and disable PEs. A mask statement of the form:

where<data condition>**do**<instructions>

would cause each PE to evaluate the <data condition>, using the data in its own memory. Only those PEs in which the condition was true would execute the <instructions>; the remaining PEs would be disabled until the next block of instructions was broadcast from the control unit. Therefore, although less time is required to generate the 8-bit code, in an SIMD implementation the time to generate the 26-bit code would be required any time even one PE needed to generate the 26-bit code.

```
compute $\overline{m}_1, \overline{m}_2, \overline{\sigma}^2$
if $\overline{\sigma}^2 \approx 0$ then code $\overline{m}_1$        /* 8-bit code */

else begin                  /* 26-bit code */

    compute $\overline{\sigma}$

    construct bit plane

    code $\overline{m}_1$ and $\overline{\sigma}$

end
```

Figure 3. Steps to code each block using variable rate scheme.

Consider the best and worst case performance of the SIMD implementation for a given percentage mixture of 8-bit and 26-bit blocks. In SIMD mode, each PE would be coding a block from the same column of the image, but from a different line group. The best case will occur when the data is such that whenever any PE generates an 8-bit code all PEs generate an 8-bit code. In this case, a factor of N speedup over the serial algorithm will be obtained. however, in actual images, this situation will rarely exist. The worst case performance will occur when the data is such that at any point in the algorithm, at least one PE is generating a 26-bit code. Let B be the number of blocks in the images, α the percent of 8-bit blocks, $(1-\alpha)$ the percent of 26-bit blocks, t_8 the time to generate an 8-bit code, and t_{26} the time to generate a 26-bit code. Then the serial execution time will be:

$$T_1 = [\alpha t_8 + (1-\alpha) t_{26}] B.$$

The worst case N-PE SIMD execution time will be:

$$T_N = t_{26} B / N$$

and the speedup will be:

$$T_1 / T_N = [\alpha t_8 / t_{26} + (1-\alpha)] N. \qquad (5)$$

Four typical aerial reconnaissance photographs were analyzed for the purpose of determining the usage of the 8-bit and 26-bit codes. Evidence from these pictures indicates that $\alpha \approx - 10\%$, and that for any given column, it is likely that at least one PE will need to generate a 26-bit code. The low variance blocks tended to occur primarily near the edges of the image, and were clustered in groups of four or five blocks. Therefore, except possibly for coding along the left and right edges of the picture, it is likely that for any given column of the image, at least one PE will be generating a 26-bit code. PEs generating an 8-bit code will be idle for the extra time required by the slower coding scheme. Despite this disadvantage of SIMD processing, the speedup for $\alpha = 0.1$ is greater than 90% of N.

In terms of speedup, the absolute worst case for the SIMD mode of processing will occur for an image where 1/N of the blocks generate a 26-bit code, and these blocks are distributed so that whenever one PE is generating a 26-bit code, all others need only to generate an 8-bit code. Under this pathological worst case percentage mixture and data distribution, speedup will be:

$$T_1/T_N \approx (t_8/t_{26})N.$$

Because SIMD processing will rarely be able to take advantage of the shorter time needed to code a block to 8 bits, MIMD operation is considered, with each PE coding the blocks in its portion of the image asynchronously with respect to the other PEs. Best case performance will occur when the low variance blocks occur in the image in such a way that they are evenly distributed among the PEs. The 8-bit blocks need not be in the same column of the image for the savings in execution time to be realized. If each PE holds the same number of 8-bit blocks, the total execution time will be the same for all PEs, and for such images, execution time of the N-PE MIMD algorithm will be faster than the serial algorithm by a factor of N. Worst case performance will occur if all of the blocks in some PE require the 26-bit code. The coding of the image will not be completed until that PE has finished; meanwhile, the remaining PEs will be idle. The worst case speedup for a given α will be given by Equation 5. Thus the best and worst case speedups will be the same as for the SIMD algorithm. However, because the requirements on the relative placement of the 8-bit blocks is less stringent for good speedup under the MIMD approach, for realistic images the MIMD algorithm will in general be faster than the SIMD algorithm. Moreover, for any given image, the SIMD algorithm will never be faster than the MIMD algorithm. Any data distribution that will improve the speed of the SIMD algorithm will also improve the speed of the MIMD algorithm. The converse is not true. For this reason, for variable rate coding, the MIMD implementation method will be preferable. The exact difference between the SIMD and MIMD execution times for a given image will be a function of factors such as data distribution, α, and system architecture implementation details.

Decoding

For decoding under the fixed rate scheme, the code is assigned to PEs in an analogous pattern to the data assignment for coding. Each PE holds code for λ line groups, corresponding to 4 λ image lines. If SIMD processing is used, each PE decodes its portion of the image, however speedup will be slightly less than a factor of N. This is a result of the fact that among those pixels being decoded simultaneously, some PEs will need to assign the output level y_2. In an SIMD system, this could be implemented using data conditional masking. In this case, a mask would be of the form

where<data condition>**do**<instructions 1>**elsewhere do**<instructions 2>

Each PE would evaluate the <data condition> using data in its own memory. Those PEs in which the condition is true would execute <instructions 1>. Then the remaining PEs would execute <instructions 2>. The time to execute the "where-elsewhere" statement would be the sum of the times to execute <instructions 1> and <instructions 2>. Therefore, instead of one output level assignment per pixel as in the serial algorithm, it will most often be the case that time for two assignments will be needed in the SIMD algorithm. If the fixed rate decoding is implemented as an MIMD process with each PE asynchronously decoding its stripe of the image, a speedup of N over the serial algorithm will be obtained. The relative speeds of SIMD versus MIMD processing for this task would depend on the actual architecture implementation details.

The decoding operations in the variable rate method are shown in Figure 4. As in the variable rate coding, the processing of 26-bit blocks requires more computation than the processing of 8-bit blocks. However, whereas the operations for coding to 8 bits were basically a subset of the operations to code to 26 bits, for the decoding process entirely different operations are performed for the two types of codes. In an SIMD system, this could be implemented using "where-elsewhere" statements. For the variable rate decoding, unless all PEs were decoding blocks of the same type, the time to decode one set of N blocks (i.e., one block per PE) would be the time to decode a 26-bit block plus the time to decode an 8-bit block. For this reason, for decoding in the variable rate method MIMD processing is preferable. Two allocations schemes are considered.

```
if number of bits = 26
    then begin
        compute output levels $y_1$ and $y_2$
        decode each pixel
    end
    else assign value of $\overline{m}_1$ to each pixel
```

Figure 4. Steps to decode each block under the variable rate scheme.

In the first MIMD decoding scheme, the coded data is assigned in a manner corresponding to the data assignment for coding (code for λ line groups per PE). Because the decoding of 26-bit blocks requires more computation than decoding of 8-bit blocks, performance will vary as it did in coding. Best and worst case performance will be analogous.

An alternative data allocation divides the coded bits (as opposed to blocks) evenly among the PEs. This scheme will reduce the possible imbalance in computation among the PEs, since PEs holding primarily 26-bit blocks will hold and decode fewer blocks than PEs holding a large number of 8-bit blocks. For example, a PE holding only 26-bit blocks will decode approximately one-third as many blocks as a PE holding only 8-bit blocks. Each PE will receive complete block encodings, i.e., the bits representing a block will not be split across two PEs. One way to accomplish this is to include with each encoded image a count of the number of 8-bit and 26-bit encodings. This information can be used to estimate the total processing time required and to guide the data allocation when the encoded image data is read into the parallel secondary storage units (or directly into the PCU memory modules, via the primary/secondary storage bus system, in a real-time application).

For images having a very irregular distribution of low variance blocks, this bit-based allocation will help to equalize the execution time of the PEs. However, the scheme may have an additional overhead of restoring the image line structure after decoding is completed. The extent of this overhead will depend on the next task for which the data will be used. If the decoded image is to be displayed on a device with a word serial input channel, there will be no additional overhead. The data from the PEs will just be read into the device in sequence, either after being written back into the secondary storage system or directly from the PCU memory modules (using the primary/secondary storage busses).

V. CONCLUSION

Ways in which the PASM parallel processing system can be used to perform two types of BTC have been presented. Both SIMD and MIMD algorithms have been considered. Due to the fact that each block of the image is processed independently, no inter-processor communication or synchronization is needed, and so, on the average, the MIMD approach will be faster than the SIMD. The parallel algorithms described here are not limited to PASM. They could be used with any SIMD or MIMD system.

Future work will include a closer look at the processing rates demanded by real-time coding/decoding at TV frame rates, and how this impacts the algorithms used by the memory management system (see Figure 2).

VI. REFERENCES

[1] E. J. Delp and O. R. Mitchell, "Image Compression Using Block Truncation Coding," *IEEE Trans. on Communications*, Vol. COM-27, No. 9, Sept. 1979, pp. 1335-1342.

[2] O. R. Mitchell and E. J. Delp, "Multilevel Graphics Representation Using Block Truncation Coding," *Proceedings of the IEEE*, Vol-68, No. 7, July 1980, pp. 868-873.

[3] E. J. Delp and O. R. Mitchell, "Some Aspects of Moment Preserving Quantizers," IEEE Communications Society's International Conference on Communications (ICC), June 1979, Boston, pp. 7.2.1-7.2.5.

[4] O. R. Mitchell, S. C. Bass, E. J. Delp, T. W. Goeddel and T. S. Huang, "Image Coding for Photoanalysis," *Proceedings of Society for Information Display*, Vol. 21, No. 3, 1980, pp. 279-292.

[5] W. H. Chen and C. H. Smith, "Adaptive Coding of Monochrome and Color Images," *IEEE Trans. on Communications*, Vol. COM-25, No. 11, Nov. 1977, pp. 1285-1292.

[6] H. J. Siegel, L. J. Siegel, F. C. Kemmerer, P. T. Mueller, Jr., H. E. Smalley, Jr., and S. D. Smith, "PASM: A Partitionable SIMD/MIMD System for Image Processing and Pattern Recognition," *IEEE Trans. on Computers*, Vol. C-30, Dec. 1981.

[7] M. J. Flynn, "Very high-speed computing systems," *Proc. IEEE*, Vol. 54, Dec. 1966, pp. 1901-1909.

[8] W. J. Bouknight, et al., "The Illiac IV system," *Proc. IEEE*, Vol. 60, Apr. 1972, pp. 369-388.

[9] K. E. Batcher, "STARAN parallel processor system hardware," *AFIPS 1974 NAT'l. Comp. Conf.*, Vol. 43, May 1974, pp. 405-410.

[10] W. A.Wulf and C. G.Bell, "C.mmp - a multi-miniprocessor," *AFIPS 1972 FJCC*, Dec. 1972, pp. 765-777.

[11] R. J. Swan, S. H. Fuller, and D. . Siewiorek, "Cm*: a modular multi-microprocessor," *AFIPS 1977 Nat'l. Comp. Conf.*, June 1977, pp. 637-644.

[12] H. J.Siegel and R. J. McMillen, "Using the augmented data manipulator network in PASM," *Computer*, Vol. 14, Feb. 1981, pp. 25-33.

[13] H. J.Siegel and R.J. McMillen, "The multistage cube: a versatile interconnection network," *Computer*, to appear.

INDEX OF AUTHORS

Abaya, E. F. 79
Agrawal, D. P. 559
Akl, S. G. 381
Alden, R. T. H. 777
Arce, G. R. 869
Arya, S. 486
Asselmeyer, B. 451
Asthana, A. 494
Baker, D. J. 412
Banerjee, U. 514
Baras, J. S. 269
Barmish, B. R. 304
Becker, R. I. 746
Belbas, S. A. 269
Bentley, J. L. 11, 364
Berger, T. 70, 404
Bertsekas, D. P. 291
Brown, C. A. 644
Brown, M. E. 534
Burke, H. H. 669
Bucklew, J. A. 60
Calahan, D. A. 486
Calise, A. J. 452
Carter, M. J. 51
Chalk, C. R. 329
Chandramouli, R. 707
Chang, S. S. L. 470
Chellappa, R. 850
Chen, C.-T. 586
Chen, C.-W. 290
Choueka, Y. 762
Chow, A. L. 214
Chung, J. C. 320
Cohoon, J. 1
Colbourn, C. J. 741
Coleman, N. 659
Comroe, R. A. 260
Cooper, G. R. 247
Costello, D. J., Jr. 260
Cristi, R. 312
Delp, E. J. 891
Dennenberg, L. 626
Dickinson, B. W. 78
Djaferis, T. E. 175
Doberkat, E.-E. 373
Dunham, J. G. 69
Dunlavey, M. R. 389
Dziwak, W. 688
El-Nahas, I. 777
Ephremides, A. 412
Eslami, M. 153
Eulrich, B. J. 329
Fathi, E. T. 524
Fenton, R. E. 804
Fine, T. L. 605
Fischer, T. R. 440
Fong, C. B. 860
Fraenkel, A. S. 762

Gallagher, N. C., Jr. 99, 869

Garzia, M. R. 430

Goodman, S. A. 688

Govindaraj, K. S. 329

Grygier, P. A. 819

Guarino, C. R. 259

Gurari, E. M. 752

Hagihara, K. 234

Hakimi, S. L. 731

Halverson, D. R. 569

Havlicsek, B. L. 721

Heald, E. J. 247

Hemami, H. 819

Hsu, Y. P. 890

Huang, H. 138

Huang, T. S. 890

Huang, Y. F. 607

Hunt, L. R. 462

Ilic-Spong, M. 786

Ito, K. 296

Ito, T. 645

Iyengar, S. V. 504

Johnsson, L. 235

Kanefsky, M. 596, 860

Kashyap, R. L. 850

Kassam, S. A. 586

Keil, J. M. 41

Khatib, R. 195

Kim, H. Y. 144

Kinney, L. L. 504

Kirkpatrick, D. G. 41

Kolb, N. 69

Konstantinides, K. 722

Krieger, M. 524

Kuhl, J. G. 553

Lakhani, G. D. 211

Lazar, A. A. 261, 279

Leathrum, J. F. 678

Lee, C.-C. 617

Lee, S. H. 461

Lescanne, P. 634

Leung, A. 829

Levine, W. S. 829

Lewis, H. R. 626

Li, H. 769

Lim, T. L. 248, 249

Liu, J.-B. 786

Lo, J. T.-H. 128

Lodaya, M. D. 153

Loh, N. K. 659

Loparo, K. A. 430

Magee, R. E. 804

Manacher, G. K. 618

Marchetti-Spaccamela, A. 31

Martin, C. F. 430

Martinez, A. B. 576

Masson, G. M. 543

McGillem, C. D. 119

Meditch, J. S. 413
Mehravari, N. 404
Meijer, H. 381
Meyer, G. 462
Meyer, G. G. L. 721
Mickle, M. H. 769
Mintz, M. 688
Monopoli, R. V. 312
Morris, S. B. 543
Mudge, T. N. 891
Mueller, M. S. 249
Munson, G. 70
Nakamura, K. 421
Nakajima, K. 697, 731
Neuhoff, D. L. 51
Ng, S.-K. 128
Nishitani, Y. 645
Nodes, T. A. 99
Okamoto, E. 421
Özgüner, F. 713
Özgüner, U. 836
Pate, E. B. 688
Perl, Y. 746, 762
Printis, R. 814
Probert, R. L. 204
Purdom, P. W., Jr. 644
Raghavan, R. 1
Rajput, A. Q. K. 596
Raman, K. V. 452
Rao, K. R. 879
Reddy, S. M. 553
Reid, R. E. 450
Roberts, C. S. 494
Rozsa, P. 163
Säck, J.-R. 21
Sahni, S. 1
Saidahmed, M. 480
Santoro, N. 195, 204
Sarachik, P. E. 797
Sato, Y. 421
Sawan, M. E. 174
Schwartz, S. C. 261
Shapiro, E. Y. 144, 320, 339
Shimizu, Y. 421
Siegel, H. J. 542, 891
Siegel, L. J. 880, 891
Simaan, M. 842
Sinha, N. K. 163, 777
Six, H. W. 348
Sloane, N. J. A. 98
Snir, M. 224
Snyder, D. L. 290
Sobel, K. M. 339
Spong, M. W. 296
Srinivasan, R. 879
Stanat, D. F. 364
Steele, J. M. 364
Stubberud, A. R. 109

Su, R. 462
Sudborough, I. H. 752
Swanson, G. K. 494
Swaszek, P. F. 89
Sziwak, W. 688
Tarn, T. J. 296
Teshigawara, Y. 421
Thomas, J. B. 89, 576, 607
Thowsen, A. 463
Tokura, N. 234
Toussaint, G. T. 21
Towsley, D. 397
Tran, M. T. 174
Tuomenoska, D. L. 542
Tugcu, A. K. 450
Tugnait, J. K. 185
Tzeng, F. T. 60
Vogt, W. G. 769
Wada, K. 234
Wegner, L. 348
Weide, B. W. 534
Weiselthier, J. 412
Willard, D. E. 355
Wise, G. L. 79, 569
Wolf, J. K. 397
Wolfenson, M. 605
Wolff, S. S. 654
Wu, K.-T. 237
Xia, G. H. 109
Yildrim, B. 836
Yoder, M. A. 880
Yu, K. B. 119
Zaborszky, J. 786
Zaghloul, M. E. 480
Zhao, G. 163
Zheng, Y.-F. 819
Zhou, Y. F. 796

UNIVERSITY OF ILLINOIS-URBANA
621.3811AL5 C001
PROCEEDINGS / ALLERTON CONFERENCE ON COM
1981
3 0112 008149160